Expressive Visualization and Rapid Interpretation of Seismic Volumes

Daniel Patel

Thesis for the degree of Philosophiae Doctor (PhD)
at the University of Bergen, Norway

August 2009

Acknowledgement

I would like to thank the people who supported me during my PhD:

To my mother and father for unlimited love. To my brother for guidance in both work and in life, for leading the way and demonstrating where not to go. To my aunt, uncle and family in Connecticut for unreserved hospitality and for offering me and my friends a peaceful place for rest and care.

To Ivan for being a fine friend and for advice and inspiration in research. To Kha for long lasting friendship, Bård for deep conversations, Eirik for supporting my decision, Silja for caring, Magnus for great fun, Katarina for sophistication, Kristina for sharing thoughts, Christian for always being there, Mathilde for being sweet, Egle for good memories, Inga-Karin for being a friend, Ulf for physical development and good conversations and Ingelin for adventures. My thanks also go to Raymond, Agathe, Stefan in Frankfurt, Halvard, Gergø and Hege.

Scientific environment

This PhD has been made possible through a scholarship from Christian Michelsen Research. I am grateful to CEO Arvid Nøttvedt and vice presidents Kåre Villanger and Christopher Giertsen for standing behind this initiative. They identified one of the leading visualization research groups in Europe; the visualization group at the Technical University of Vienna, where I spent about half the PhD under my thesis supervisor Meister Eduard Gröller. Thanks Meister and CMR for excellent supervision and for giving me freedom to develop my own ideas.

The thesis topic of illustrative visualization was directly inspired by the work of Stefan Bruckner, Ivan Viola and Peter Rautek in the Vienna group. Much knowledge and fun came from them and the other group members: Erald Vucini, Peter Kohlmann, Martin Haidacher who coauthored a paper, Muhammed Muddasir Malik, Maurice Termeer and Matej Mlejnek. The fine VolumeShop software developed by Stefan Bruckner has mainly been the development framework for the techniques presented in this thesis.

The other half of my PhD was spent at the then newly created group at the University of Bergen now led by Professor Helwig Hauser and Ass. Prof. Ivan Viola. I had several good discussions with the professors and the PhD students Johannes Kehrer, Ove Daae Lampe who is also sponsored by CMR and Paolo Angelelli. The PhD student Jean-Paul Balabanian who coauthored a paper helped me a lot with ideas and support. The work by the Master student and coauthor Øyvind Sture has also been very valuable in my PhD.

Without guidance from domain experts in the oil and gas industry my research would not be possible. These people work on very tight schedules, so thanks to my co-supervisor Dr. Christopher Giertsen for identifying and getting me in contact with the following experts in StatoilHydro who were willing to discuss new ideas: Thanks to Dr. John Thurmond for listening to ideas and coming back with real scenarios and data to demonstrate them on. Thanks to the geologist and illustrator John Gjelberg for giving me a glimpse into the art of illustration drawing and for showing how a 'geoillustrator' does his work. Thanks to Jens Grimsgaard for supporting the project.

List of publications

Illustrative Rendering of Seismic Data
Published in: Vision, Modelling and Visualization (VMV) 2007
Authors: Daniel Patel, Christopher Giertsen, John Thurmond, Eduard Gröller

The Seismic Analyzer: Interpreting and Illustrating 2D Seismic Data
Published in: IEEE Transactions on Visualization and Computer Graphics, vol. 14, no. 6, November/December 2008
Authors: Daniel Patel, Christopher Giertsen, John Thurmond, John Gjelberg, and Eduard Gröller

Moment Curves
Published in: Proceedings of Pacific Vis, April 2009
Authors: Daniel Patel, Martin Haidacher, Jean-Paul Balabanian, Eduard Gröller

Seismic Volume Visualization for Horizon Extraction
Technical report/manuscript, 2009
Authors: Daniel Patel, Stefan Bruckner, Ivan Viola, Eduard Gröller

Knowledge-assisted visualization of seismic data
To be published in: Computers and Graphics Journal
vol 33 issue 4, scheduled to appear end-September / beginning-October 2009
Authors: Daniel Patel, Øyvind Sture, Helwig Hauser, Christopher Giertsen, Eduard Gröller

List of presentations

Paper I presentation: Saarbrücken, Germany, November 7, 2007

Knowledge-assisted Visualization Workshop 'Knowledge-assisted visualization of seismic data', IEEE VisWeek, Colombus, Ohio, USA, October 19, 2008

Paper II presentation: IEEE VisWeek, Colombus, Ohio, USA, October 23, 2008

Arranger/chair on half day session 'State of the art Visualization in Geosciences'.
European Geosciences Union, General Assembly 2009, Vienna April 20, 2009
(Self)invited talk: 'Illustrative Visualization for Rapid Interpretation in Geosciences'

Paper III presentation: Pacific Vis, Peking University, Beijing, China, April 23, 2009

Invited talk: IllustraVis Workshop, University of Bergen 'Illustrative Visualization for Rapid Interpretation in Geosciences', June 4, 2009

Invited talk: VRGeo Consortium, Fraunhofer IAIS, 'Illustrative Visualization of Seismic Data', Sankt Augustin, Germany, June 16, 2009

Contents

Chapter 1

Abstract

One of the most important resources in the world today is energy. Oil and gas provide two thirds of the world energy consumption, making the world completely dependent on it. Locating and recovering the remaining oil and gas reserves will be of high focus in society until competitive energy sources are found. The search for hydrocarbons is broadly speaking the topic of this thesis. Seismic measurements of the subsurface are collected to discover oil and gas trapped in the ground. Identifying oil and gas in the seismic measurements requires visualization and interpretation. Visualization is needed to present the data for further analysis. Interpretation is performed to identify important structures. Visualization is again required for presenting these structures to the user. This thesis investigates how computer assistance in producing high-quality visualizations and in interpretation can result in *expressive visualization and rapid interpretation of seismic volumes.* Expressive visualizations represent the seismic data in an easily digestible, intuitive and pedagogic form. This enables rapid interpretation which accelerates the finding of important structures.

Chapter 2

Introduction

The diverse work of the five articles constituting this thesis has been divided into the categories of expressive visualization and rapid interpretation. Expressive visualization is one of the two enabling technologies for rapid interpretation. The other enabling technology is automatic structure identification algorithms. Such algorithms segment data into meaningful objects. Certain expressive visualizations makes use of segmented data by applying different presentation styles on each object in the data. In the geoscientific domain, segmentations are typically obtained through an interpretation process. Thus it is not a strict ordering with visualization first and then interpretation. Visualization and interpretation goes in a spiral. Visualization enables interpretation, which leads to more information about the data that can be used to produce better visualizations. This in return gives more insight into the data.

2.1 Expressive Visualization

Expressive visualizations present data in a more expressive and clearer way than standard visualization methods such as slice rendering, polygonal surface rendering and volume rendering do. Expressive visualizations are perceptually more easy to digest. This is achieved by shaping the presentation of the data to how the human perceptual system works. Previous studies and knowledge from art, illustration techniques and perceptual psychology has been used to develop methods which efficiently present data in a communication friendly form. We are applying these methods to create expressive visualizations. Illustrations are examples of expressive visualizations. Illustrations are frequently used for communicating completed seismic interpretations. The ability of illustrations to express abstractions make them well suited for representing the high-level knowledge that is produced during interpretation. In the oil and gas industry, high-quality illustrations have a central role in the final communication of interpretation results for decision making and in reporting. The expressive visualizations presented in this thesis attempt to:

- reduce the time to create geoscientific illustrations.

- make it easier for interpreters to express their hypotheses and inner models.

- offer new visualization methods tailored for seismic data.

The three bullet points are in order discussed in the next three sections.

Computer-generated Illustrations Geoscientific illustrations are ideal for communicating interpretations to decision makers and to the public. They are extensively used in reports when bidding for concessions to perform surveys and drills. The illustrations are also important in educational material for geosciences.

Automated and tailored tools for creating geoscientific illustrations are lacking. The creation process is time consuming and manual. Manually drawing illustrations binds valuable human resources in an oil company as it requires special competences from geologists with illustration knowledge. We attempt to increase the efficiency of geoscientific illustrators by automating common and time consuming drawing tasks such as sketching textures and drawing suggestive horizons. These tasks are currently performed rather manually in general purpose drawing software. In contrast, automating this procedure results in the generation of illustrations in fractions of a second after a model is defined. The sub-second rendering time gives rise to a new mode of interactivity in illustrations. Tasks such as changing the angle of the illustration, the perspective, the texture types or the cut planes takes place with immediate results. This is an important improvement from manual illustrations where performing such changes requires drawing a new illustration and may take hours or days. Interactivity in illustrations increases their communicative impact compared to classical static illustrations.

Hypothesis Externalization Instead of only applying the illustrative techniques on finished interpretations, we have considered the effects of using computer assisted illustrative sketching techniques *during* interpretation. The quality of seismic data can vary, the data can be noisy and unclear. Consequently the interpreters must compose hypotheses of possible scenarios and make educated guesses. Interpretation is regularly performed in multidisciplinary teams and it is important that the members understand each others ideas. By allowing quick sketching of illustrations through computer assistance, the interpreters are given a tool for expressing and sharing their internal hypotheses and models. Externalizing hypotheses provides the means for discussing ideas and arriving at a common ground of understanding.

Improved Seismic Visualization The visualization research communities have had little focus on seismic data. Promising results have been achieved within subjects such as medical visualization and information visualization. We attempt to reduce this gap by adapting and tailoring methods from other visualization fields and developing new visualization methods tailored for seismic data such as:

- improved 3D seismic visualization using real-time approximated ambient occlusion volume rendering.
- 3D sparse texturing for conveying rock layering and rock types in true 3D.

2.2 Rapid Interpretation

Interpretation is required when searching for oil and gas. Oil and gas is created when organic material is deposited, buried and pressurized under high temperatures over long periods of time. The oil and gas will then migrate upwards and may get trapped in tightly sealed reservoir structures and accumulate there. For finding the oil or gas, signs of the producing deposits or collecting seals are searched for with interpretation techniques. The aspects of rapid interpretation in this thesis attempt to reduce the manual interpretation work and increase interpretation speed by:

- providing overviews of the data so interpretation in unimportant areas is avoided and focus can be made on areas with potential.
- reducing time spent on repetitive and time consuming tasks by automating them so the interpreter can concentrate on challenging areas in the seismic data where automatic methods fail.
- making it quicker and easier to verify a finished interpretation by comparing it and visualizing it aligned and in context with the underlying data the interpretation was based on.

These points of rapid interpretation are further elaborated in the three following sections.

Overviews One element enabling rapid interpretation is the ability to gain an overview of the data. Interpretation is typically time consuming due to focus on high accuracy at the start. The reason for high accuracy at the very start stems partly from the limited possibilities to create overviews in this early stage. Overviews are difficult to obtain because of the large seismic volumes collected from extensive land or sea areas. Several physical measurements for the same area also cause obtaining overviews challenging. Overviews are accomplished in this thesis with methods for compact visualization of large areas and with methods for the simultaneous visualization of several measurements in overlapping areas. Large areas are visualized by constructing sparse representations containing only the absolutely necessary information and presenting this information in zoomed-out thumbnail views. Overlapping volumetric measurements are presented by mapping the volumes to disparate representations that are perceptually separable, followed by merging the representations carefully into one view. This procedure provides the means for multiattribute visualizations.

Automatic Structure Identification Another element enabling rapid interpretation is having automatically extracted structure suggestions at hand before manual interpretation begins. This allows the interpreter to quickly browse through the suggestions and select good ones. In contrast, in the current workflow the interpreter is either manually identifying and marking structures or, when working in a semiautomatic mode, the interpreter is constantly interrupted by the initialization of structure identification algorithms followed by waiting for the automatic processing to finish.

Verification Interactivity in illustrations allows easier verification of the interpretation represented in the illustration. In an interactive rendering the user can switch from seeing the original uninterpreted data to seeing the interpreted illustration. Smoothly fading out the original data while fading in the illustration makes it possible to compare the underlying data with the interpretation and this enables verification.

Chapter 3

Results

This thesis consists of 4 published articles and one unpublished technical report. Each article emphasizes parts of the elements described in the introduction. Using the same order and structure as the introduction, this chapter presents the results we have achieved. References to the articles for further reading is given.

3.1 Results in Expressive Visualization

The techniques for constructing expressive visualizations have been developed through the study of subjects such as art, illustration techniques and human perception. The following works give a broader overview of the underlying literature and science behind creating expressive visualizations. 'Semiology of graphics' [5] written by Bertin in 1967 can be considered a classic work laying the foundation of information visualization. Bertin studied the visual symbols used in drawings and graphics such as lines, patterns, stipplings and colors. He showed how these symbols could be combined to achieve information-rich visualizations. Tufte wrote the book 'The Visual Display of Quantitative Information' [14] about efficient presentation of information. He takes numerous examples of good and bad charts and diagrams from newspapers and analyses them. Guidelines for efficient visual communication of information is presented in his works. A scientific approach towards perception is taken by Margaret S. Livingstone in the book 'Vision and Art: The Biology of Seeing' [13]. Livingstone is a neurobiologist, she explains why the brain more easily understands certain visual representations over others and makes connections to techniques used in art.

Before we present our results in expressive visualizations we start with describing the seismic data and how it can be visualized using basic techniques. The data we are visualizing in this thesis is mainly seismic reflection data. This data is acquired by sending sound waves into the earth and processing the reflected sound. This is analogous to how ultrasound measurements are performed in medicine. A basic seismic survey results in a 2D image of a vertical slice into the ground. More advanced 3D surveys result in a series of 2D vertically stacked slices spanning a 3D reflection volume of the earth.

From the seismic reflection data, new data can be calculated. The new data will emphasize a certain property or *attribute* of the original data. Since the new data is *derived* from the original, such data is called *derived attributes.* Further information about seismic attributes can be found in the work by Iske and Randen et al. [11].

Visualizing data requires a mapping from the numerical data values to the 2D grid of colored pixels constituting the computer display. The visualization of a 2D seismic slice is created by mapping from the measured slice values to colors. Several color mappings can be used. For seismic reflection data it is common to map the range of low-to-high reflection values to a color transition from red via white to blue or from black to white, see Figure 1. The look-up table that defines the mapping from data values to a visual presentation is called a transfer function. See Figure 2 for a 3D seismic reflection dataset rendered with a red-white-blue transfer function and Figure 3 top left where a white-black transfer function is used. The transfer function can also assign degrees of transparency to the value range. Transparencies result in blending and visualization of the data values behind the transparent ones, this creates a 3D rendering of the volume. Hadwiger et al. [10] give an in depth introduction to the principles of volume rendering. To enable the appearance of objects in 3D space as if the computer screen is a look-through window into a 3D world, translations and rotations are performed on the data followed by a projection to the 2D computer screen. The reader is referred to Foley et al. [8] for basic concepts of transformations and the rendering pipeline.

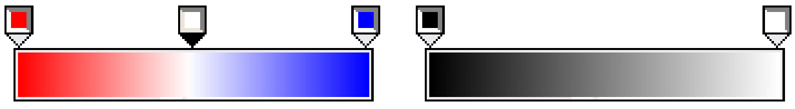


Figure 1: Red-white-blue and black-white transfer functions

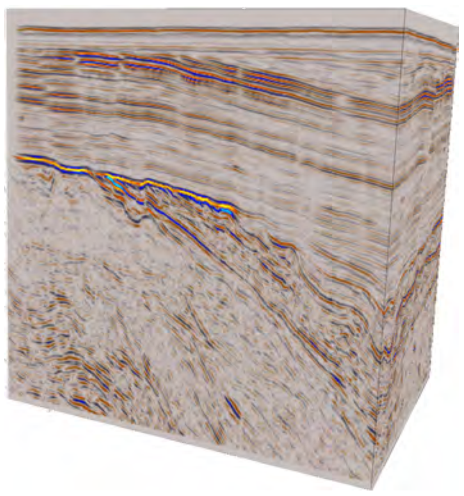
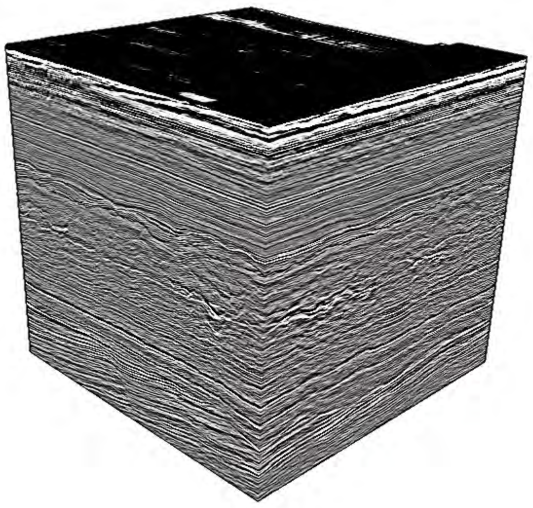


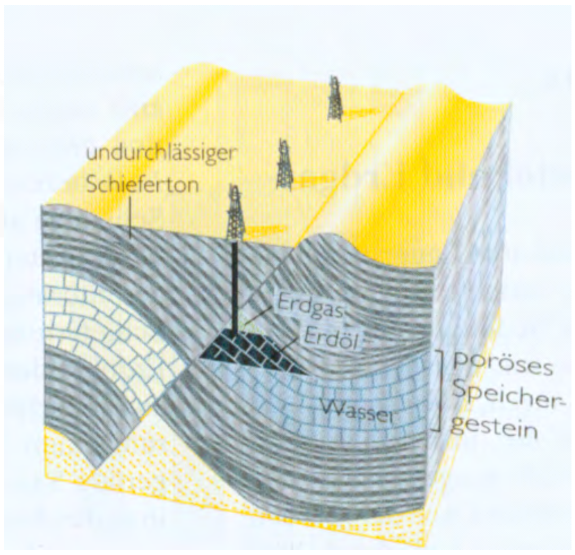
Figure 2: Red-white-blue seismic reflection volume. No reflection values have been set to transparent, therefore the volume can not be looked into and only the side surfaces of the volume block is visible.

Basic visualization techniques were briefly discussed in this section. The following section will present more advanced visualization techniques conceived in this thesis.

3.1.1 Computer-generated illustrations

The drawing techniques for creating illustrations have been developed for the purpose of communicating high-level aspects in a simple way. Elements of interest are illustratively emphasized to represent the knowledge acquired during some analysis process. Illustrative methods allow for annotating the data so that the resulting image communicates its intent clearly to the viewer. Figure 3 gives a comparison of computer-generated raw data visualization in a) vs a manually crafted illustration in b). Figure 3c) is a computer-generated result from this thesis' work.


(a) Raw seismic data



(b) Manually drawn illustration

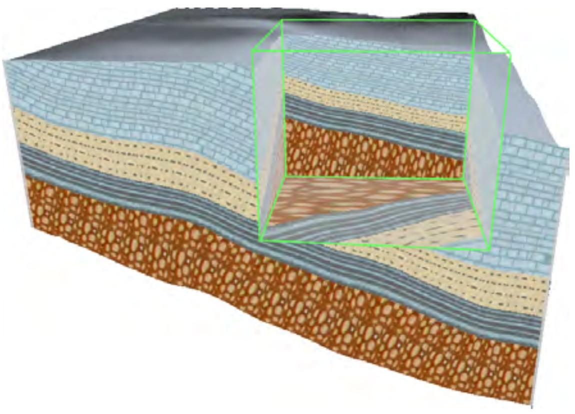

(c) Computer generated illustration

Figure 3: a) Raw data. b) Manual illustration from 'Understanding Earth' [9] showing an oil and gas seal made from a faulted impermeable rock layer. c) Our computer-generated illustration where the content of the green wireframe box has been removed to enable insight into the volume.

As can be seen in the handcrafted illustration taken from geoscientific literature, textures are used extensively. There are several reasons for this. As opposed to color-coding, textures are perceptually more expressive. Textures communicate the rock type and the rock orientation. Compression can be presented by varying the density of the texture throughout the layer. Erosion and faults can be shown using discontinuities in the textures (a fault is shown as the diagonal crack in Figure 3b). Information is embedded directly in the texture. In addition, textures give aesthetically pleasing results. Reading out information without having textures requires more work. Imagine Figure 3b) without textures thus only having line borders separating the layers and the fault. Identifying the rock type would now require searching through the layer for a text label or for a line going to a text label describing the rock type. Even if the layers would be color-coded the user would still have to look up the rock type in a color legend. The angle of the rock at any position would now be deduced by finding and averaging the angle of the upper and lower boundary line of the layer close to that position. Compression would be deduced by looking at the varying distance of the upper and lower boundary line. Thus without textures, the information is no longer locally presented but must be searched for in the image. For interactive visualizations further problems arise when trying to deduce layer angles and compression if the layer boundaries are not visible on screen due to high zoom levels.

These are some reasons why the use of textures was adopted and integrated into the geosciences. To ease communication, geologists and geoillustrators use a standardized language for expressing their knowledge. This language consists of different textures for representing information such as rock types. The US Federal Geographic Data Committee has produced a document [2] with over one hundred of such standardized textures. Figure 4 shows three pages from this geological texture library. The patterns are cleverly designed

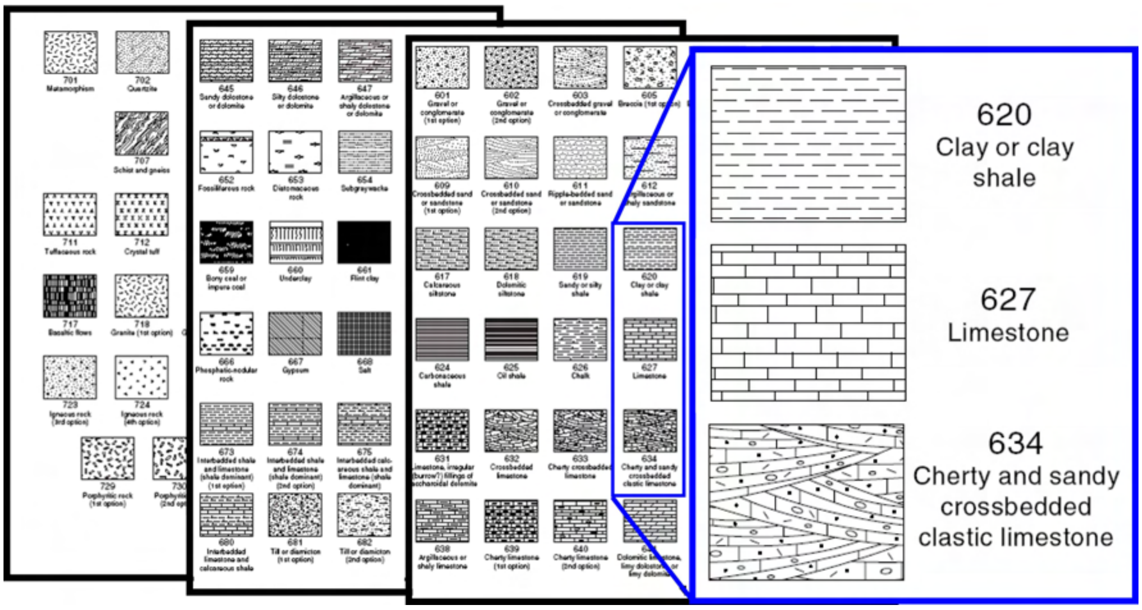


Figure 4: Common symbology in geosciences. Rock textures [2].

and constitute a taxonomy. Rock types of the same subgroup are represented with similar patterns. Thus, even if a geologist does not recognize the exact pattern she is looking at, she should be able to identify the general group it belongs to when knowing other rocks with similar patterns.

On earth, rock layers are laid down in succession and they represent successive geological time periods. A widely used and standardized table of color-codings representing geologic time from the current Quaternary period to the Archean eon 2.5 billion years ago is also part of the 'visual' or 'symbolic' language used by geoscientists (see Figure 5). This language has been developed by the Commission for the geological map of the world [1].

In contrast to other domains where advanced visualization is frequently used such as in medicine, geoscientists heavily make use of a standardized visual language for communication. Therefore expressive visualizations such as illustrations have a large potential in the geoscientific domain. One of the goals of this thesis has been to automate the techniques required for creating geoscientific illustrations and integrate the use of illustrations

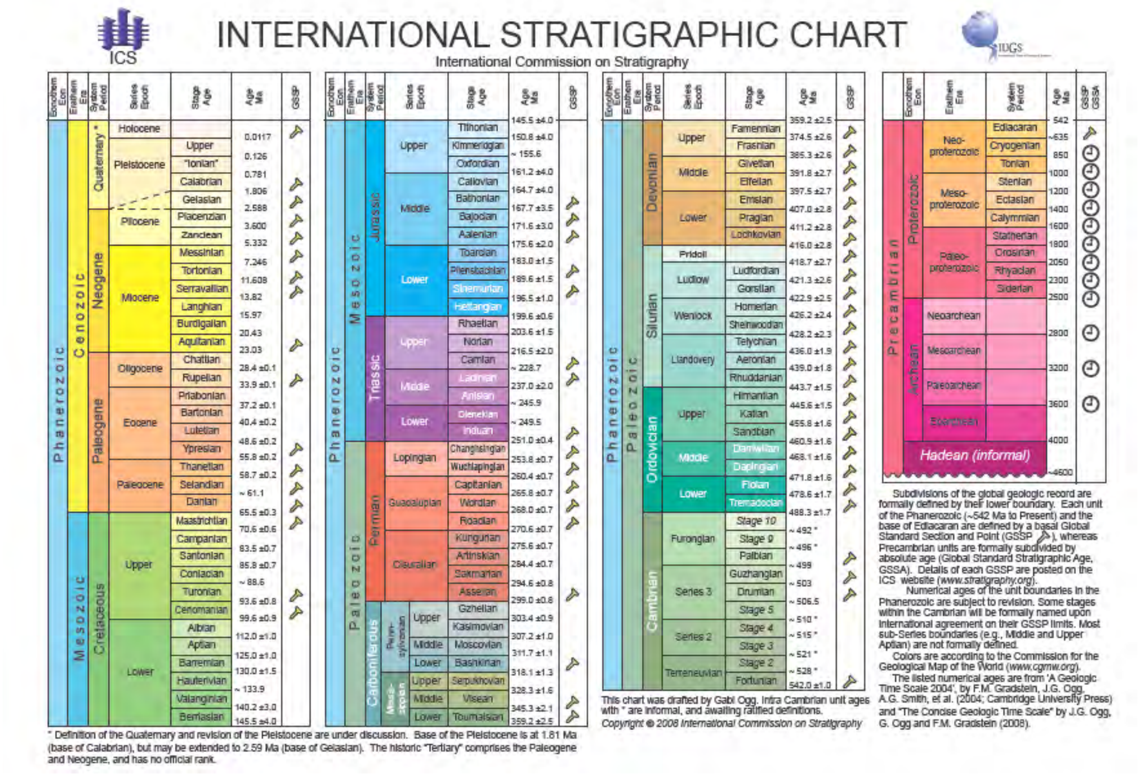


Figure 5: Common symbology in geosciences. Geologic time colors [1].

in the work flow. Achieving the expressive visualizations that these illustrations represent, requires two components. One must assign textures to the data and one must specify how the textures will bend. These two components are elaborated on in the following two sections.

Texture Transfer Functions In our work we achieve the texturing seen in illustrations by using transfer functions extended to handle textures. We call these 'texture transfer functions'. Similar to color transfer functions which assign colors to attribute values, texture transfer functions assign textures to attribute values. Figures 6a) and b) show examples of two texture transfer functions to the left and the respective results when applied on the seismic data to the right. The textures shown in squares above the transfer functions to the left blend into each other analogous to how the colors in Figure 1 shown above the transfer functions blend into each other. In the simplified examples of Figure 6, the horizontal axis of the transfer function is not mapped to a seismic attribute, but is simply mapped to the horizontal axis of the result image to the right. Therefore textures to the left in the texture transfer function are seen to the left of the result image and textures to the right are seen to the right in the result image. In addition to assigning textures, transparencies are assigned along the horizontal axis of the transfer function. The transparencies are defined by the height of the blue curve in the transfer function. When the curve is at its vertically lowest in Figure 6 left, the texturing is completely transparent and when the curve is at its vertically highest, the texturing is completely opaque. By using transparencies it becomes possible to look through the texture and directly at the data displayed behind. By varying how textures and transparencies are assigned to the data values, several effects can be obtained. Figure 6a) shows a transfer function that softly fades in from complete transparency where the underlying seismic data is visible, to complete opaqueness and then blends between textures. Textures bend according to the underlying seismic data displayed above the result images. To achieve this effect the data must be analyzed so the 'bending' can be extracted and applied to the textures. Extraction of the bending information is described in the next section. In Figure 6b) an abrupt change from transparent to opaque takes place due to the staircase shape of the opacity curve in the transfer function. The opaque textures then blend from few random lines to textures with an increasing number of lines. In example b) the textures do not bend according to the underlying seismic as this is not appropriate for all types of textures.

Figure 7 shows a slice through a dataset manually segmented into four rock layers. The layer segmentation is the same as shown in Figure 3c) although other textures are used. Each rock layer is assigned a unique texture. Two textures controlled by an attribute is shown on top of the rock layer textures. They are a boulder texture (blue) for low values and a brick texture (yellow/pink) for high values of the attribute. The high valued interval represented with the brick texture is further differentiated by blending in a yellow-to-pink color transfer function. This distinguishes between areas of low values in yellow, to high

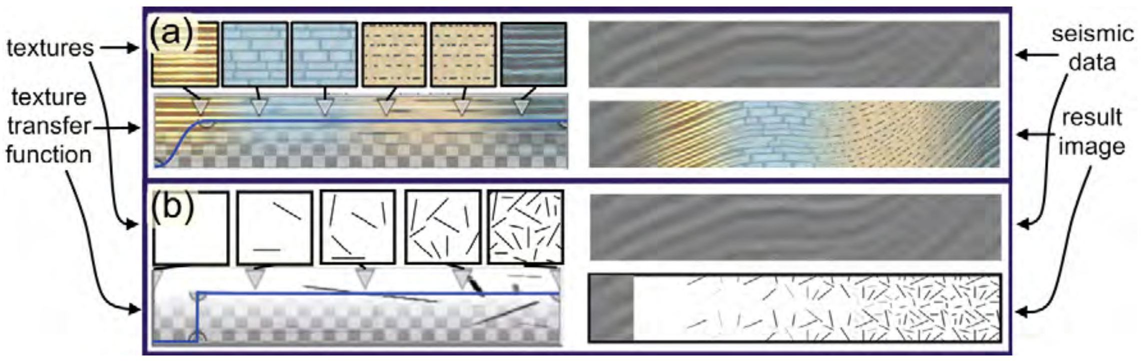


Figure 6: Two texture transfer functions left in a) and b) and the results to the right. The slices in the top right corner of a) and b) show the underlying seismic data.

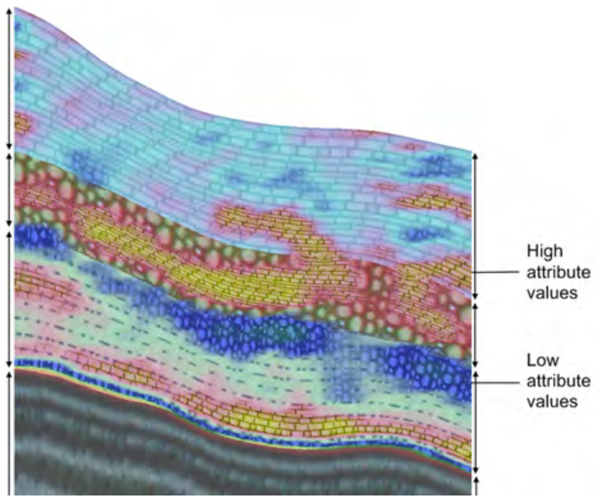


Figure 7: Color and texture transfer functions combined. Arrows denote layer extents.

values in pink for the value interval. Thus the high valued interval is expressed with a specific texture and the value range in this interval is shown with color. Color and texture can be visually mixed and then perceptually separated due to their perceptual orthogonality. The example demonstrates how the use of color and texture transfer functions can control the amount of information that needs to be shown in different parts of attribute value ranges. However the resulting visualization suffers from visual overload and it is difficult to see the underlying rock layering.

Figure 8a) shows another approach for combining colors and textures. A slice with three rock layers is shown. In the middle layer, four intervals of a derived attribute is presented. The derived attribute is shown in Figure 8b) with a color transfer function dividing the attribute values into low (blue), middle (yellow), high (red) and highest (green) values. Mapping the four intervals to different textures would hide the texturing that communicates the rock type for each layer as was shown in the previous example. To keep the rock texture for each layer, all intervals in the middle layer are mapped to the same brick texture in Figure 8a). The intervals are separated by having differently sized textures. The sizes are chosen so the four intervals are with increasing values mapped to increasingly smaller brick textures. The result is brick sizes giving an intuitive visual ordering of the four intervals while at the same time not hiding the layer texturing. The color transfer function used in Figure 8b) is blended in to further contrast the intervals. As can be seen in this example, texture type and texture size can be considered perceptually orthogonal. More examples of texturing for multiattribute visualization will be presented

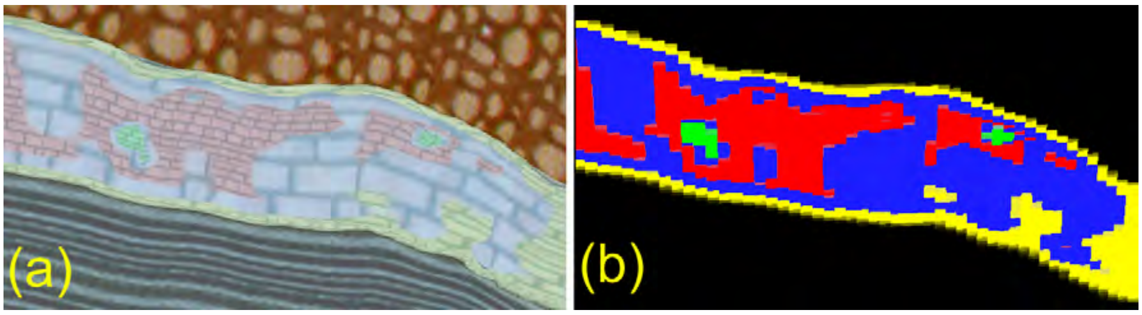


Figure 8: Color and texture transfer functions combined with varying texture density.

in section 3.2.1.

Parameterization To achieve bending textures as found in illustrations, we need to extract bending information from the seismic data, store the information in an appropriate representation, and use the information to deform the textures. The information describing the bending is encoded in a parameterization. One way to represent the parameterization

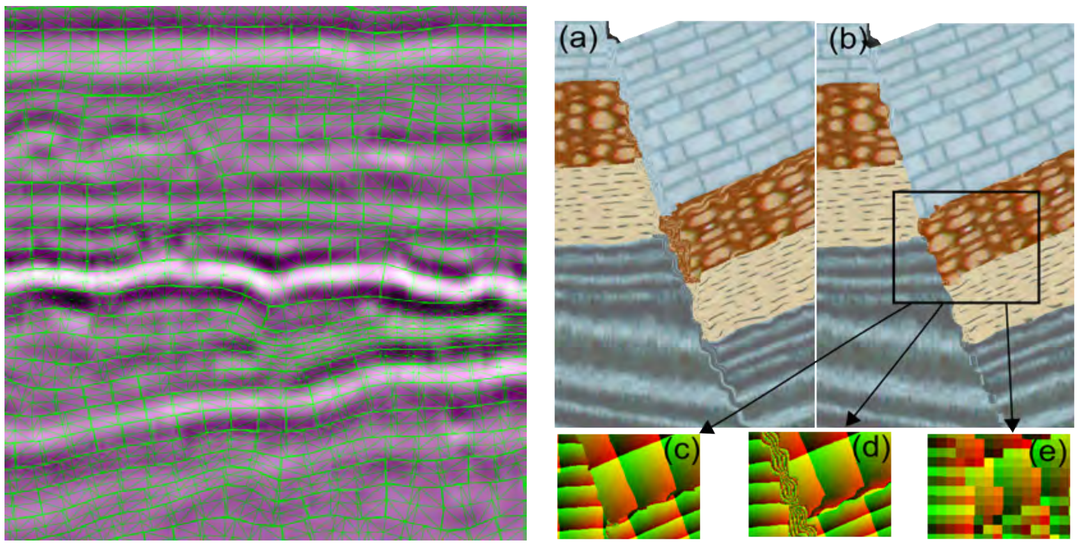


Figure 9: Comparison of parameterization methods of Paper I to the right and Paper II to the left represented as the green overlaid grid.

is by using a grid that is warped to the trend of the seismic data as seen in green in Figure 9 left. The grid spans a deformed Cartesian coordinate system. Drawing textures in this coordinate system will achieve the wished texture bending. A stylized example of Figure 9 left is shown in Figure 10a). Two red lines exemplify a structure in the reflection data and a deformed grid is shown that follows the bending and contraction of the structure. The deformed grid of 10a) is undeformed in Figure 10c). Drawing an undeformed texture in this undeformed Cartesian coordinate system is trivial. With the pair of deformed and undeformed grids, a mapping is created that defines the parameterization. This is the representation that was used in Paper II.

A parameterization representation inverse to the one just described was used in Paper I. Here the parameterization is stored in a regular grid. Basically, for each position of a volumetric reflection value, an additional value is stored describing which coordinate from the texture to use there. In Figure 9b) can be seen a texturing and in 9e) is seen a regular grid of colored pixels representing texture coordinates . This is exemplified in Figure 10b) where a regular grid is shown and in 10d) where the texture lookup coordinates for the corresponding grid points are shown. The approach has weaknesses. A problem arises when trying to represent discontinuous texture mappings such as over the fault in Figure 9b). Linearly interpolated texture coordinates stored in a regular grid disallows discontinuous texture mappings. The discontinuous texture mapping over the fault must be represented by a discrete value jump in the texture coordinates. Due to linear interpolation, a continuous value interpolation between unrelated texture coordinates takes place instead. The result of this erroneous linear interpolation of 9e) is shown in 9d). The method makes it impossible to represent discontinuous texture mappings which are needed when texturing

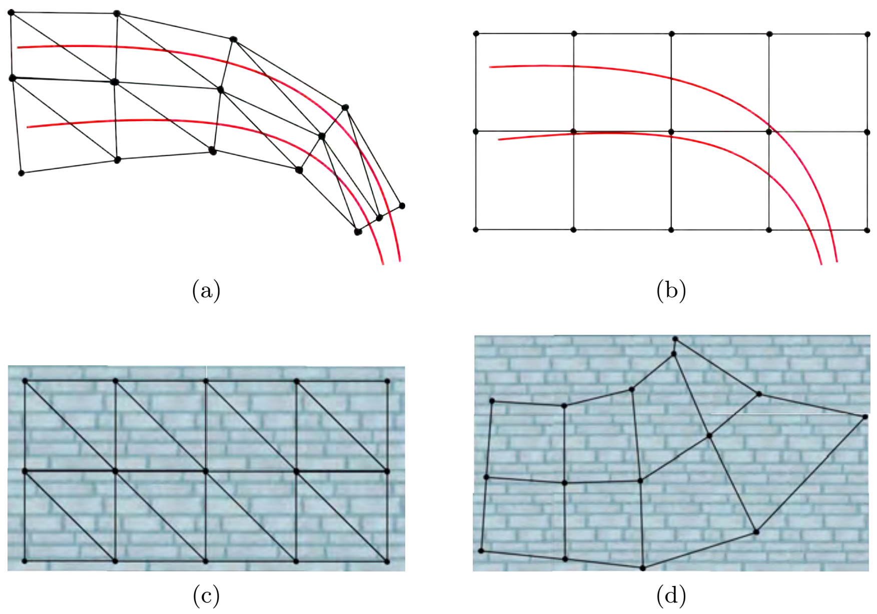


Figure 10: Conceptualized comparison of parameterization methods of Paper I in a) and c) and Paper II to in b) and d).

over faults. The textural artifacts created over a fault with this method are shown in 9a). In 9b) and 9c) our nonoptimal attempt to solve these problems is shown. Further information of this parameterization can be found in Paper I. Since the solution was not optimal, the better parameterization representation first described was developed for the follow up paper (Paper II).

Two possible representations of the parameterization were discussed above. There are also several ways to extract the parameterization from the underlying data. Structures spanning the data is required for dictating the parameterization. In our works these structure have either been manually interpreted horizons or horizons automatically extracted from the uninterpreted reflection data. Paper I uses manually interpreted horizon surfaces to create a 3D parameterization. In Paper II we create a 2D parameterization using horizons lines automatically extracted from uninterpreted 2D data. A natural next step would be creating a 3D parameterization from uninterpreted 3D data. We believe that the distance transform from the extracted 3D horizon patches used in Paper IV can be a promising starting point for this.

3.1.2 Hypothesis Externalization

Expressive visualization techniques can be used to create geoscientific illustrations that convey finished interpretations. Creating illustrations from completed interpretations has been the current work mode for geoscientific illustrators. We have investigated the effects of illustration sketching early in the workflow, while interpretation is being performed, instead of after it is finished. Such an approach can be useful for communicating ideas and hypotheses during interpretation. This can be an advantage as compared to only having expressive visualizations available after both the interpretation is finished and the manual illustration has been made. An example of a hypothesis quickly sketched during interpretation is shown in Figure 11. Different textures have semitransparently been overlaid on a seismic slice to express one possible subdivision into rock layers. Using our techniques, such an illustration is created in a matter of minutes. It can be used to express an interpreter's internal idea of the layering. When the expressive illustration is made, his ideas are more easily grasped by others. More about such illustrations and Figure 11 in particular can be read in Paper V.

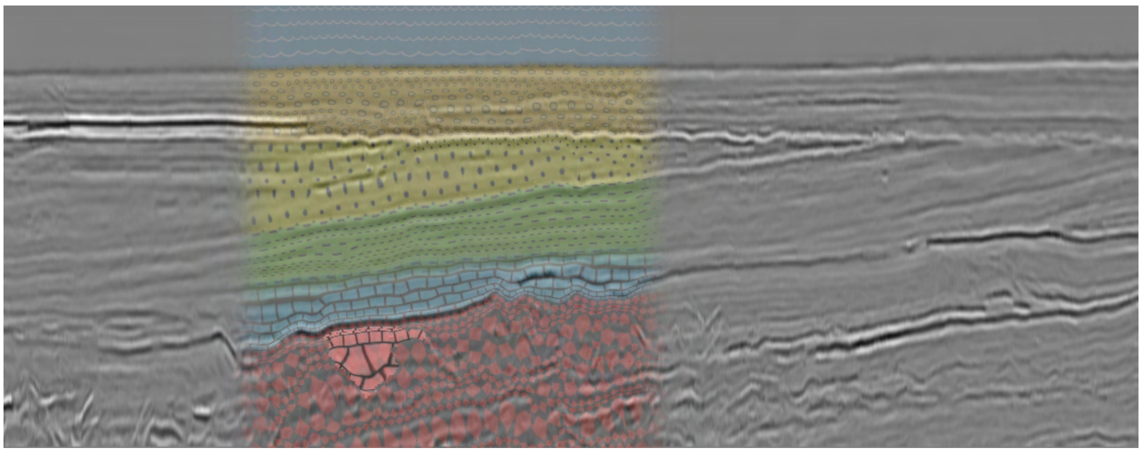


Figure 11: Seismic data overlaid with textures representing a hypothesized rock layering.

3.1.3 Improved Seismic Visualization

In the two next sections results are presented achieving improved seismic visualizations by adopting more realistic lighting models during volume rendering and results for 3D texturing.

Ambient Occlusion In Paper IV we introduce a new volumetric rendering method for seismic data based on ambient occlusion. The method is inspired by optical realism and achieves more natural shadowing, better depth perception of the data and allows for glowing effects to emphasize important regions. A major challenge for providing useful 3D interactive visualization is the choice of an appropriate 3D rendering algorithm. Seismic data are typically noisy and lack distinct material boundaries as the acquisition is based on acoustics. The widely used gradient-based methods as introduced by Levoy [12] are in general sensitive to high-frequency noise. Gradient-based shading of seismic data introduces distracting artifacts which makes interpreting the 3D renderings difficult. Other approaches, such as unshaded direct volume rendering or maximum intensity projection [15] tend to depict seismic data as a homogeneous cloud without distinct features. Thus, common approaches are frequently unsuitable for visualizing seismic data. Until now only unshaded direct volume rendering and gradient-based methods have been used for 3D seismic data. We have identified a volume rendering technique that is promising for seismic data which has not been able to be rendered in real-time until recently. The method is an approximation of ambient occlusion rendering. The approximation enables interactive rendering of the volume with real-time modifications of the transfer function. See Figure 12 for a comparison of gradient-based (left) and ambient occlusion based (right) rendering.

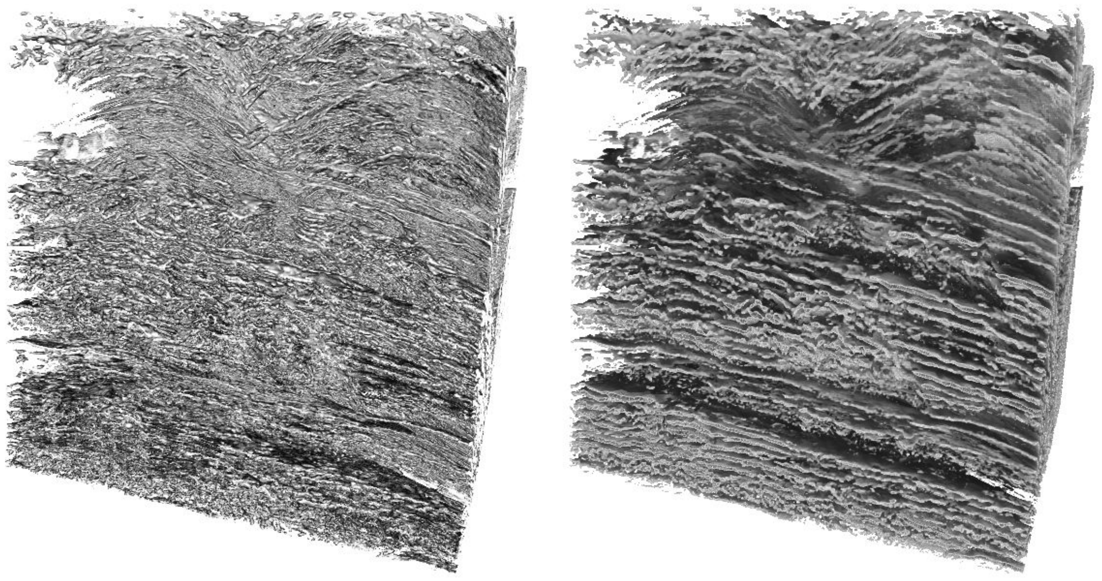
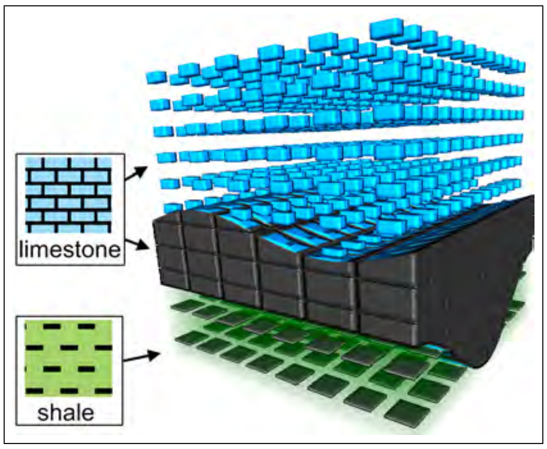
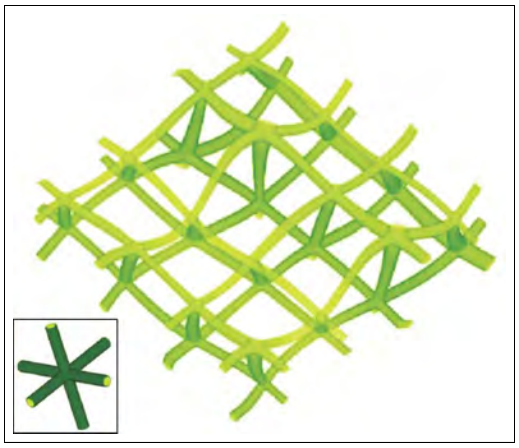


Figure 12: Standard gradient-based rendering to the left and our real-time approximation of gradient-free ambient occlusion to the right.

3D Sparse Texturing Paper V discusses the possibilities to extend the widely used 2D texturing in the geosciences to 3D by using sparse 3D textures. All standardized geologic textures are defined in 2D (Figure 4). However the seismic data to be interpreted is inherently 3D. 2D textures lend themselves directly to illustrate 2D seismic slices but have limitations when applied on 3D data. The current solution for 3D data is to apply 2D textures on planar cross sections. This technique is seen in 3D geological illustrations such as in Figure 3b). Several advantages can be gained if suitable 3D textural representations of the 2D textures are specified and used. From an algorithmic point of view it is simpler to map 3D textures to 3D volumes and surfaces than 2D textures. Distortion problems arise when mapping 2D textures to curved surfaces, and frame-to-frame incoherencies arise when interactively moving a 2D textured cut-plane. 3D textures will reduce these problems of spatial and frame-to-frame incoherencies. Using 2D textures on semitransparent volumetric regions is not well defined. Here, 3D semitransparent textures may give rise to a higher perceptual depth. Volumetric variations can be revealed inside the 3D texture and not only on the exterior boundary as is the case when using a 2D texture. However since the textures used for conveying knowledge in geology are only defined in 2D, there exist no corresponding 3D versions. If we want to use 3D textures, we must synthesize them ourselves from the 2D cases which is an underdefined task. Figure 13a) shows two 2D textures to the left and how they can be synthesized into 3D textures and used in a 3D rendering. For the limestone texture two alternative 3D textures are shown, a sparse texture at the top and a dense texture below. By also using parameterization information, the textures can be deformed and will communicate the deformation in 3D. Figure 13b) gives an example of a deformed 3D texture situated in a rock layer. The 3D texture exemplar is shown in the black rectangle in the lower left corner. The topic of 3D sparse texturing is discussed at the end of paper V.



(a) Undeformed 3D textures synthesized from 2D textures.



(b) Deformed 3D texture. The 3D texture exemplar is in the bottom left corner.

Figure 13: 3D sparse texturing allows seeing into the volumetric data.

3.2 Results in Rapid Interpretation

Building on the techniques reviewed in the previous sections and on automatic interpretation algorithms introduced in this section, methods for rapid seismic interpretation are presented here.

3.2.1 Overviews

Techniques for scale invariant and multiattribute visualizations result in overviews. Overviews can quickly identify where to focus the interpretation. Or they may early reveal if a seismic prospect has no potential and should be discarded. Either way, time is saved.

Scale Invariant Zooming Figure 14 gives two examples, to the left and right of the vertical line, of what we have called 'scale invariant' zooming. An image contains a certain amount of information (top row Figure 14). Information is averaged away or compressed beyond comprehension when simply zooming out an image. For efficient presentation, different zoom levels require different abstracted visualizations. We achieve this with our scale invariant zooming techniques. We first extract an abstraction of the information to be visualized, we use precalculated horizons and the parameterization for this. Then the information is presented at varying sparseness degrees by visualizing an appropriate amount of information for the current zoom level. Left shows normal zooming with the

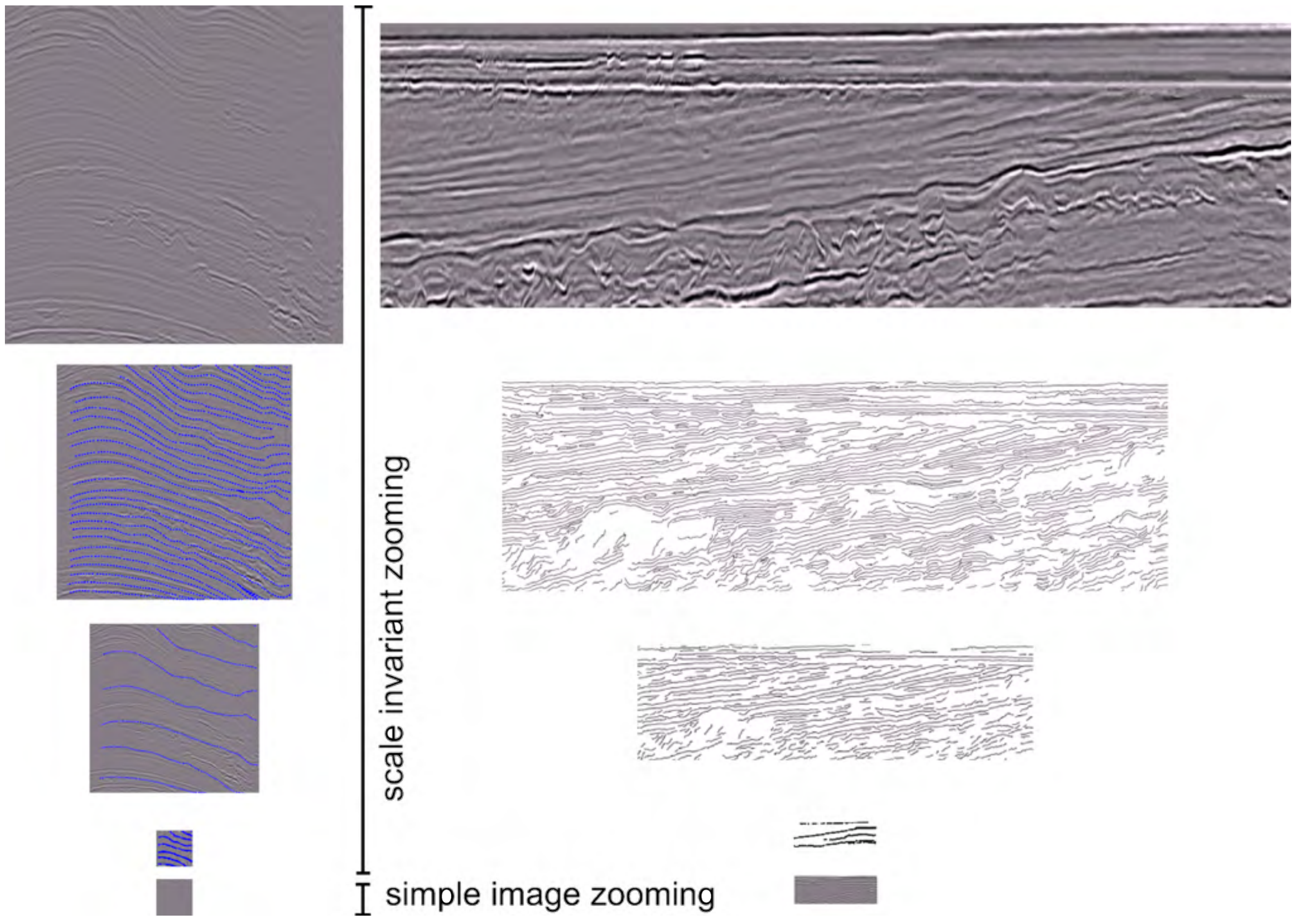


Figure 14: Examples of adaptive visualizations appropriate for different detail levels.

sparse information overlaid in blue while right shows only the sparse information. The bottom row shows the original data from the top row zoomed out in the normal way. In the bottom row literally all information is gone and only a gray rectangle is left.

Scale invariant zooming can also be used when zooming in on data. Using the method of texturing for instance, even in extreme sub-sample magnification, when zooming beyond the resolution of the data, textures will look smooth and will express the angle and the type of layer that is being zoomed in on. On such sub-resolution scales, color transfer functions would yield uninformative single colored results. The techniques for scale invariant zooming are described in Paper II.

Multiattribute Visualization Multiattribute visualization deals with visualizing several measurements in the same image. In many domains, such as medical visualization, climate visualization, or flow visualization, disparate measurements exist for the same area. In medical visualization one might have CT, MR, and PET volumes of the same organ. Being able to create a visualization that merges these volumes into one meaningful image is the topic of multiattribute visualization. For climate visualization, one has the challenge of displaying measurements such as temperature, wind direction, wind speed, and cloud factor on a geographic map already full of information. Extensive research has been done in multiattribute visualization. Bürger et al. [7] provided a state-of-the-art report. The challenge in multiattribute visualization is to merge all the information in one comprehendible image. The image can be made comprehendible to the viewer by mapping the attributes to representations that are perceptually separable from each other. More about perceptual orthogonality in textures and patterns can be read in Bertin [5]. In the seismic domain one must deal with data such as overlapping volumetric attributes, well logs and various formats that interpreted data is represented in, e.g., geometric surfaces.

Figure 15a) shows a slice of seismic reflection data. Derived from this data are the attributes chaos, dip, frequency and reflection intensity (Figure 15b - e). A multiattribute visualization of these attributes is shown in Figure 16. The reflection intensity attribute is depicted with a line transfer function. The line transfer function represents data by using curves following the parameterization which are colored according to the attribute value. Another line transfer function is used to draw stippled blue lines following the trend of the data. More can be read about line transfer functions in Paper II.

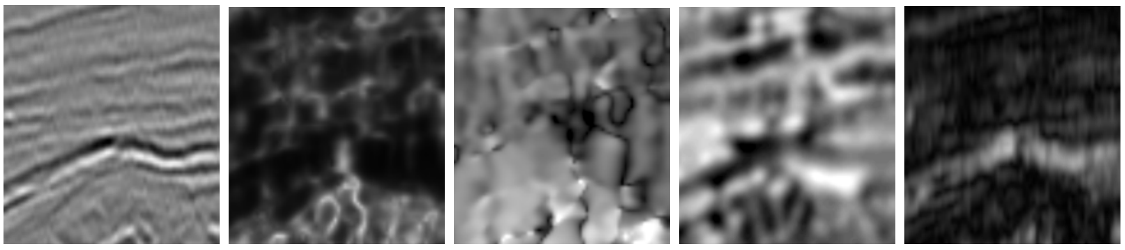

(a) reflection data (b) chaos (c) dip (d) frequency (e) reflection intensity

Figure 15: Reflection data in a) followed by attributes derived from it in b)-e).

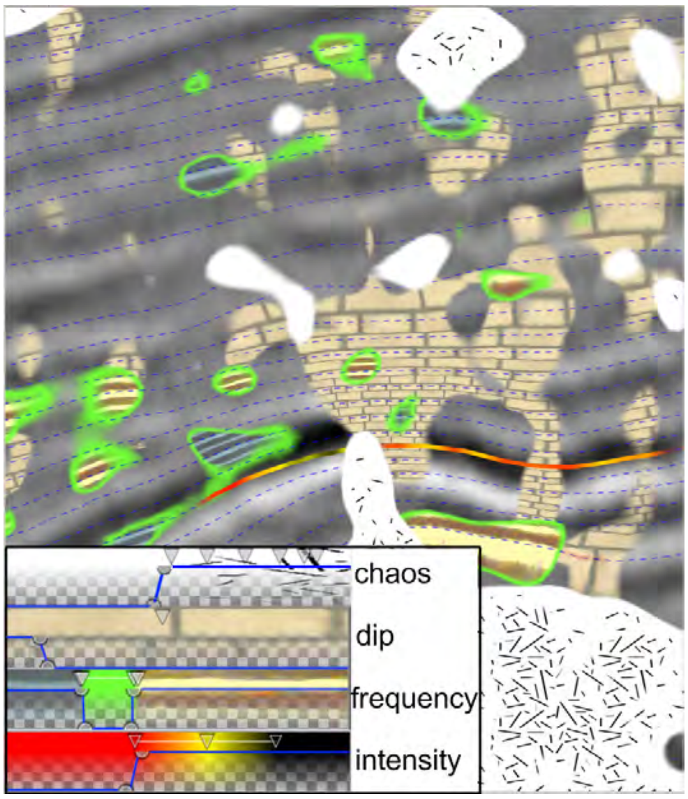


Figure 16: A multiattribute visualization with the attributes chaos, dip, frequency, and reflection intensity overlaid on seismic reflection data. The transfer functions are shown in the lower left corner. Notice the low and high frequency values with differently colored textures and a green halo effect easily obtained with the transfer function.

Figure 17 a)/d) is an example of using two differently colored versions of the same texture to encode intermediate (brown) and high (blue) dip values. High chaos values are overlaid semitransparently with an increasing amount of lines representing increasing chaos. Figure 17 b)/e) and c)/f) apply highly transparent and thus less disctracting textures. The transparency makes it possible to see the underlying reflection data. In example c)/f) small peaks in the transfer function opacities define the texture borders by making them more opaque so the interior regions can be made even more transparent.

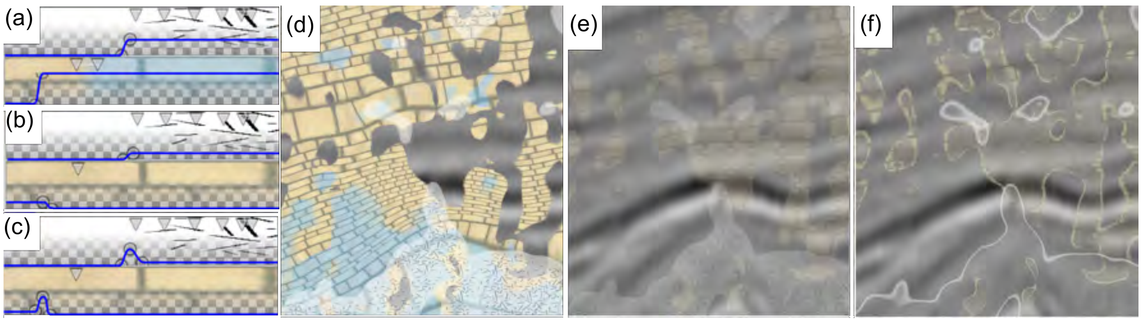


Figure 17: In a), b and c) three transfer function setups are shown each consisting of two transfer functions. The top transfer function maps chaos and the bottom one maps dip as given in Figure 15. Their corresponding renderings are shown in d), e) and f).

3.2.2 Automatic Structure Identification

The subsurface of the Earth consists of material layers with distinct densities and porosity characteristics. Horizons are central structures in interpretation defined as the interfaces between these layers. We have chosen to focus on seismic horizons since these are typically the first structures to be interpreted. They are also some of the simplest structures to identify using image processing techniques. This is due to their well defined visual appearance in the seismic data. With our methods we aim to increase the computer's assistance in finding horizon structures. Thereby enabling rapid interpretation.

Horizon Identification Computer-assisted interpretation of horizons is done by first automatically identifying horizon candidates in a preprocessing step and then presenting the candidates to the user through an intuitive user interface. The preprocessing step and the user interface for horizon selection is described in Paper II for the case of 2D slices. Preprocessing and interaction is extended to 3D in Paper IV.

By considering the reflection values of a 2D seismic slice (Figure 18a/b) as height values in a terrain (18c) one can identify the horizons as valleys and ridges (18d). We automatically trace out the valleys and ridges and create connected curves from the traces.

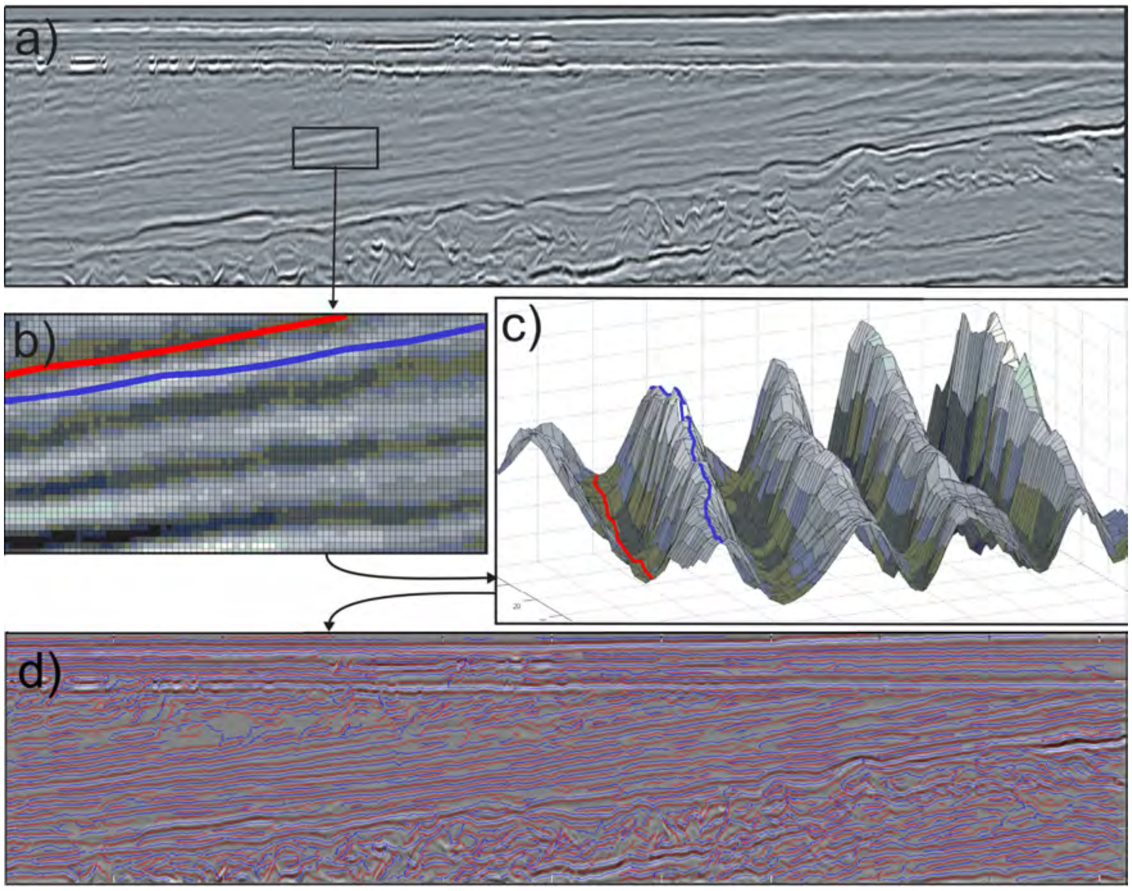


Figure 18: The 2D horizon extraction algorithm tracing along ridges and valleys in the reflection data). In c) a height field of the rectangle in a) is shown. A ridge is marked with red and a valley is marked in blue in b) and c). All extracted ridges in red and valleys in blue are overlaid on the reflection data from a) in d).

We extended the 2D horizon algorithm into 3D to enable rapid interpretation of 3D seismic horizons. A naive extension of the 2D method into 3D did not succeed. During horizon growing, too many unrelated horizons merged into single horizon candidates. After preprocessing a complete seismic volume, this could result in a single merged structure consisting of the union of all horizons found in the data. To resolve the problem, we performed a splitting of the produced horizon candidates using a hierarchical mesh-subdivision method [3]. There are many ways to split up a surface into smaller ones. Our subdivision was steered by maximizing the flatness of each surface part. After subdivision, the split horizon parts are selected in real-time by the geoscientist and assembled together into more appropriate horizons. The preprocessing steps consisting of horizon growing and subdivision are shown in Figure 19. For more information see Paper IV.

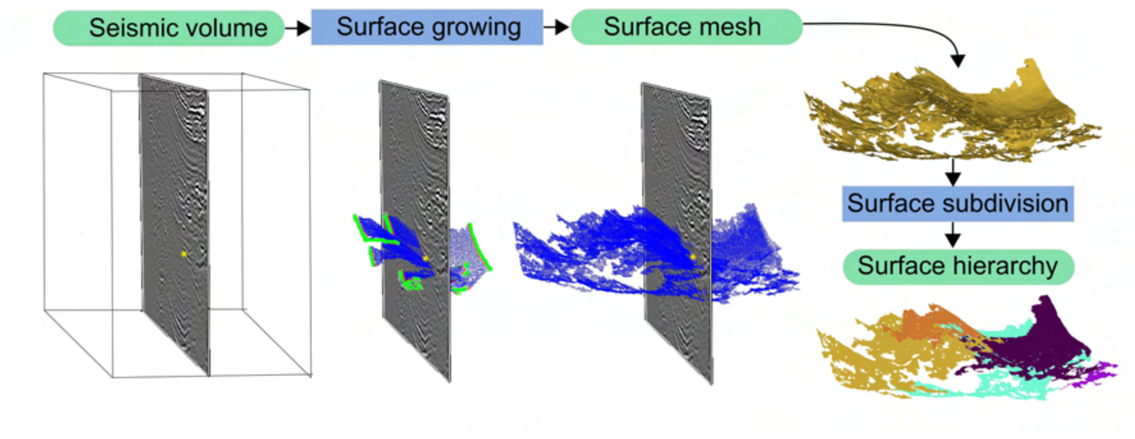


Figure 19: 3D horizon extraction followed by surface subdivision. Three steps of horizon growing from one seed point (yellow) is shown to the left. Bottom right shows the subdivision for one of the hierarchy levels.

Structure Identification by Similarity Search Instead of tailoring the structure identification algorithm for horizons only, a general method for extracting arbitrary structures based on similarity was attempted as inspired by the work of Borgos et al. [6]. Our method works by letting the user select a point on a seismic slice. Then the system shows all other points that have a similar vertical neighborhood to the selected point. The underlying assumption is that certain structures such as horizons, have distinct neighborhoods. Thus by selecting a point on a horizon, all other points on the horizon would be identified due to their distinct and similar neighborhood to the comparison point. The method works by evaluating each sample point based on its local properties only. Therefore this method is parallelizable as opposed to the sequential method of tracing horizons along ridges and valleys described in Figure 18. We were able to implement a version of the algorithm on the highly parallel GPUs of modern graphics cards. Thereby real-time performance is achieved and preprocessing is avoided.

The neighborhood that is compared is the n vertical samples above and below a specific location called comparison point. Thus n defines the extent of the neighborhood and is chosen by the user. A closeness score to the selected point is calculated for each sample in the seismic slice. This results in a new derived attribute which we call similarity attribute. A color transfer function is used on the similarity attribute and the result is overlaid on the original slice. Two examples are shown in Figure 20. The two examples have the same comparison point positioned on a horizon, but different color transfer functions. The comparison point is shown as a red dot. Its neighborhood values are shown as a yellow graph with sample values mapped to the horizontal axis. To verify that the horizon has a consistent neighborhood, the neighborhood graphs of four other points on the horizon shown in green are plotted in black. The extent of the neighborhood is shown as a colored vertical line through each point. The similarity or closeness metric has a large effect on the results. Our distance function is the sum of the squared distances between neighborhood components. We use this simple function that is quick to evaluate for achieving interactivity.

The method did not perform as well as we had expected. It was not possible to pinpoint the structures we selected with sufficient accuracy. Either a subset of the structure was selected (undersegmentation) or too much was selected (oversegmentation). Figure 20 a) shows a transfer function with transparencies set for maximizing the selection of the interesting horizon while minimizing the selection of other structures. The structure is undersegmented at the same time as areas not part of the structure are selected. Figure 20 b) shows a transfer function with more opacity defining a less strict similarity criterion. In this example we increased the transfer function opacity until the whole horizon was colored. But now oversegmentation is extensive. Thus capturing the horizon with this method is approximate only. The neighborhood along a horizon (using our metric) varies too much. At the same time due to the large amount and variation of other horizons, unrelated horizon fragments are matched. Figures 20 a) and b) show that the approach with vertical neighborhoods based on Euclidean distances is not able to identify horizons uniquely.

While trying to improve the method we noticed that the result is very sensitive when using other angles than a vertical neighborhood line. Using an angle that is normal to the horizon would better capture its neighborhood and could lead to a more consistent signature. Several heuristics with varying qualities exist for identifying the normal of a seismic horizon from the reflection data. Instead of using any of these, we extended the horizon signature to a version that was rotationally invariant by considering a spherical neighborhood instead of along a vertical line. The new method had similar problems as the

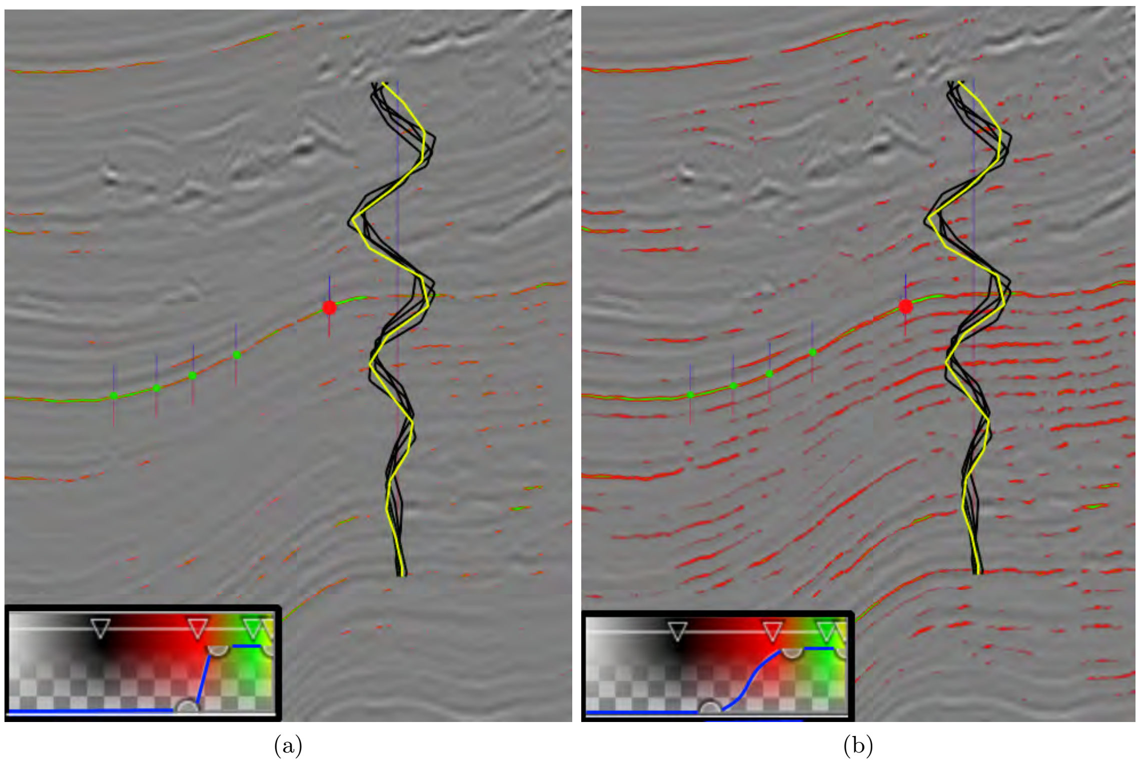


Figure 20: Horizon extraction by neighborhood similarity. The horizon intersecting the red dot is attempted extracted. Its vertical neighborhood intensities are plotted in the yellow graph. Intensities of the green dots are plotted in black graphs. In a) a strict similarity criterion defined by the color transfer function in the lower left corner is used. In b) a less strict similarity criterion is used. Oversegmentation is visible in both examples.

previous one. However it proved itself useful on other modalities which will be discussed in the next section.

Moment Curves Although not directly related to seismic data, this work originated from the rotational invariant signature on seismic data just described. By considering the evolution of the mean and variance in a spherically growing neighborhood around the sample positions we were able to achieve promising segmentations of CT and MR data. Each voxel in a volume is assigned a sequence of the mean and variance values of the voxels in a spherical growing neighborhood. We then assign optical properties to the voxel based on this sequence of values. This results in a novel classification of materials and material boundaries. The evolution of the mean and variance as the spherical radius around a voxel increases, defines a curve in 3D for which we identify important trends and project it back to 2D along the radius axis. The resulting 2D projection can be brushed for easy and robust classification of materials and material borders. See Figure 21 for a 2D projection of the 3D curves. Figure 23 shows brushing and the corresponding classification on two coronal slices through a male CT dataset. The blue dots in Figure 21, the characteristic archs and why only the interior of the organs are classified in Figure 23 is explained in Paper III.

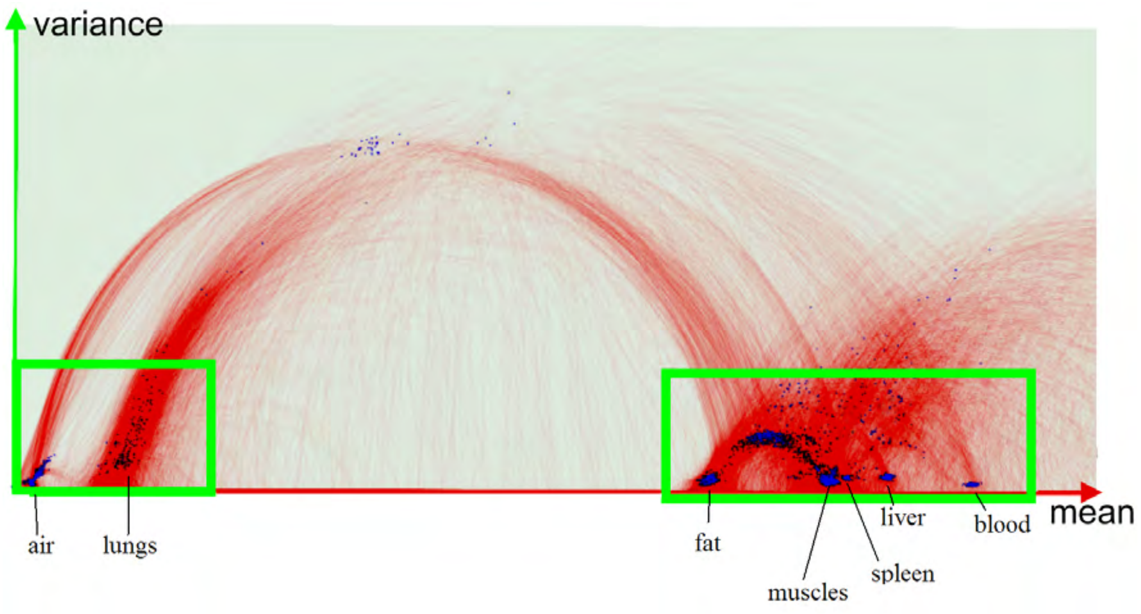


Figure 21: In red is seen (mean,variance,radius) curves projected into the mean/variance plane.

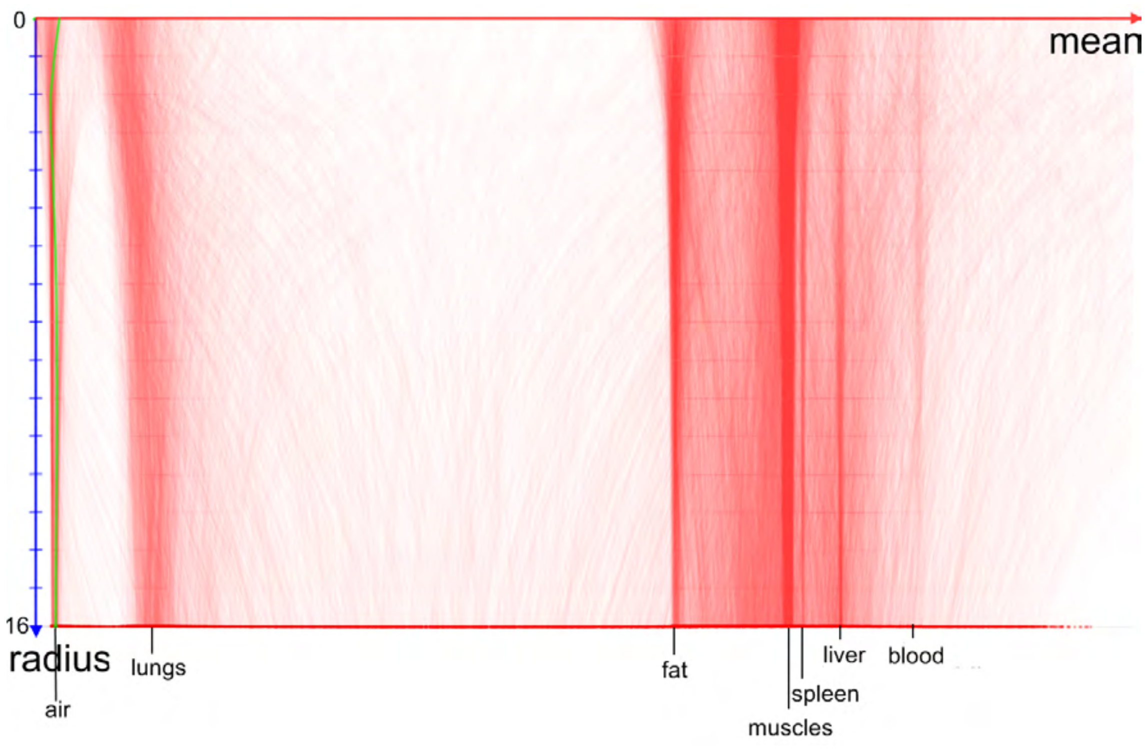


Figure 22: The (mean,variance,radius) curves projected into the mean/radius plane.

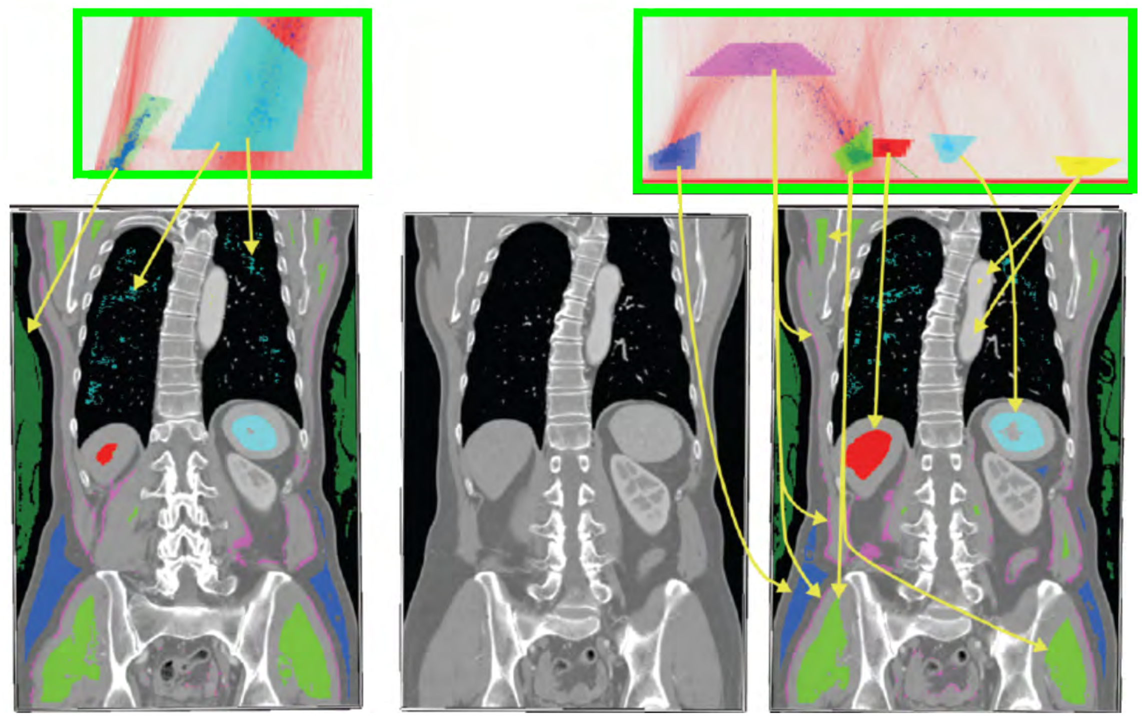


Figure 23: Brushing in the two indicated green regions from Figure 21 is shown. Bottom left and right shows two slightly different CT slices and their classified regions. Middle shows an unclassified version of the right slice.

3.2.3 Verification

Finally our methods can be used to verify a finished (Figure 24) or an ongoing interpretation hypothesis (Figure 25) with the underlying data. This is achieved by using transparency to smoothly move from the uninterpreted data to the interpreted expressive visualization as seen in Figure 24. The approach allows for comparison of the underlying data with the interpretation and thus enables verification. In Figure 25 a suggested layer subdivision has been overlaid the seismic data with high transparency. The layer borders are drawn opaquely.

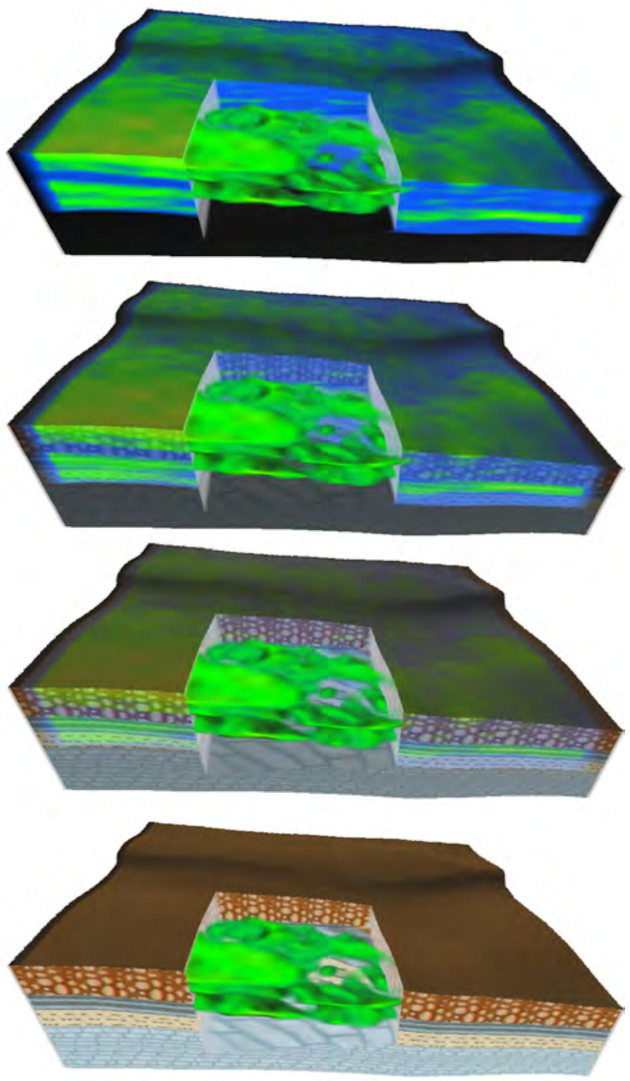


Figure 24: Verification of a finished interpretation. The top figure shows the 'impedance' attribute. Volume rendering of the attribute is shown in the cutout. At the bottom is shown an interpretation of layers.

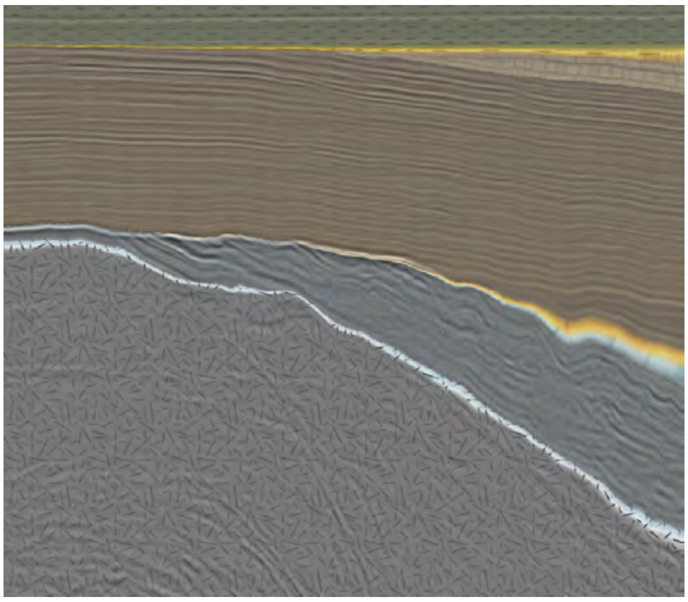


Figure 25: Verification of an ongoing interpretation.

Chapter 4

Conclusions

In this thesis we showed that geoscience has a well developed and established symbolic language for representing geo-information and that the use of manually crafted illustrations is an important part of the field. Methods for automatic generation of geoscientific illustrations were established. These methods can reduce the time to create illustrations which are used in reports and in education. The fact that computer-generated illustrations can be interacted with in real-time opens up for a more pedagogic presentation than with hand drawn static illustrations. Quickly generated illustrations also makes it possible to externalize hypotheses, making it easier for interpreters to communicate their hypotheses and inner models.

We indicated that rapid interpretation can be achieved using our expressive scale-invariance visualizations. Scale-invariant and multiattribute visualization techniques can provide overviews of the data. With such overviews one might be able to avoid interpretation in unimportant areas and spend more time interpreting areas with potential.

We presented methods for automatic structure identification of seismic horizons and, as a side track, of human tissue. Time spent on repetitive and time consuming tasks can be reduced with the automatic methods so the interpreter can focus on challenging areas in the data where automatic methods fail.

New ways of visualizing 3D seismic data were presented. Our real-time approximated ambient occlusion rendering gives enhanced visualizations of challenging high frequency and noisy seismic data. We presented the potential for 3D sparse texturing to convey rock layering, rock types, and deformations in true 3D. We also presented methods for verifying a final interpretation by seamlessly visualizing it with the underlying data the interpretation was based on.

The following paragraphs highlight observations made in this thesis.

The right level of automation A recurring theme of the thesis has been the strategy of identifying and automating work that computers can perform so humans may focus on what they do best. Here it has been important to strike the right balance by, on the one side not having an overautomatized process where the user will question and not understand the computer's suggestions, and on the other side not underautomatize the process and burden the user with time consuming monotonous tasks. We believe that the combination of computerized brute force preprocessing and a simple and responsive user interface, where the preprocessed proposals are presented to the user, helps achieve this goal. High-level preprocessing attempts to simulate human expertise. Perfect simulation of human expertise is impossible so errors are made. It is therefore important that the user can easily avoid selecting the wrongly generated suggestions with an efficient user interface.

The power of abstraction Another important point in the thesis is the use of preprocessed information that describes the structure of the seismic data. This information is of higher semantic level than derived seismic attributes. Derived seismic attributes show different properties of the seismic data but they do not give higher-level insight. For instance, in derived attributes there is no grouping of samples into objects. Higher-level information is required for further analysis and advanced visualization. We use the higher-level horizon and parameterization information for texturing, to create abstracted views of the data such as the different detail views in scale-invariant visualization and for rapid interpretation of horizons.

By randomly changing parameters and observing how this affects the visualization, the user in effect browses the reflection data and gets a better understanding of it. It is a common procedure to perform this browsing by altering view parameters and transparency parameters of the transfer function. However to get a structural overview, this real-time parameter modification has not been possible because of the need of manual intervention to extract structures such as horizons. Using the preprocessed information, the user can now quickly change parameters (such as the horizon transfer functions described in Paper II) that affect the visualization of structures and get a deeper overview .

The merge of analysis and report creation The thesis has focused on the workflow in geosciences consisting of the collection of 3D seismic reflection data, seismic interpretation, and visualization. Several concepts from this thesis can be adapted to other domains that also have these three phases, stated in more general terms, as data collection, analysis for finding results, followed by report making for communicating the results. Any domain with such stages might adapt methods presented in this thesis for the following advantages. During analysis, expressive sketching possibilities can help brainstorming and hypothesis making of scenarios. When expressive sketches made during analysis are of a quality comparable to illustrations found in reports, the tasks of analysis and report making are in effect merged. Thus the report is created continuously during analysis and not started from scratch when the analysis is finished. With expressive electronic sketches, the analysis stage is no longer a purely mental stage or a stage only semi-documented with rough pen and paper sketches. The analysis stage is better documented since the expressive visualizations represent the gained knowledge more explicit than rough sketches do.

The future - Interactive Reports Development in hardware is changing the physical appearance of reports. Computers, displays and interaction devices are merging and shrinking in size. In recent years we have seen the transition of computing power from stationary computers to laptops, netbooks and mobile phones. The size of the electronic devices reports are presented on are approaching the size of printed paper documents. We might be seeing the start of technology that will allow touch sensitive bendable electronic paper with integrated wireless network.

Development in software is changing the static nature of reports as demonstrated in this thesis. Currently, reports are typically static documents whether displayed on screens or on paper. There are increasingly examples of dynamic documents as interactive web content. A first step towards interactive reports with expressive visualizations of 3D data can be seen on the web page made by Jean-Paul Balabanian [4]. In his web page, instead of displaying a static image of a human skull, the image is generated in real-time from the underlying 3D CT scan of a head. The data can be interacted with by rotation and by changing a transfer function. Thus any aspect of the underlying head data can be visualized interactively directly in the document.

Using the new hardware and software technology together one can envision a change from static documents to dynamic and interactive hand held documents. This can lead to a new generation of reports with interactive illustrations. By integrating the underlying data with the analysis software into the report, all steps of the analysis can be gone through or even corrected. Thus analysis and report reading can be merged. We therefore conclude: In the future, data analysis and report creation will merge and all steps will be accessible and modifiable in the resulting physical document.

Bibliography

[1] *Commision for the geological map of the world.* http://ccgm.club.fr/index.html, 2006.

[2] *Federal Geographic Data Committee, Digital Cartographic Standard for Geological Map Symbolization.* Document FGDC-STD-013-2006, 2006.

[3] M. Attene, M. Spagnuolo, and B. Falcidieno. Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer*, 22(3), sep. 2003.

[4] J.-P. Balabanian. Online realtime volume raycaster. `http://www.ii.uib.no/vis/team/balabanian/project/ORVR.cgi`, 2008.

[5] J. Bertin. *Semiology of graphics.* University of Wisconsin Press, 1983.

[6] H. G. Borgos, T. Skov, and L. Sønneland. Automated structural interpretation through classification of seismic horizons. In *Mathematical Methods and Modelling in Hydrocarbon Exploration and Production*, pages 89–106. Springer Berlin Heidelberg, 2005.

[7] R. Bürger and H. Hauser. Visualization of multi-variate scientific data. In *Euro-Graphics 2007 State of the Art Reports (STARs)*, pages 117–134, 2007.

[8] J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes, and R. L. Phillips. *Introduction to Computer Graphics.* Addison-Wesley Professional, 1993.

[9] J. Grotzinger, T. H. Jordan, F. Press, and R. Siever. *Understanding Earth.* W. H. Freeman and Company, 1994.

[10] M. Hadwiger, J. M. Kniss, C. Rezk-Salama, D. Weiskopf, and K. Engel. *Real-time Volume Graphics.* A K Peters, 2006.

[11] A. Iske and T. Randen, editors. *Atlas of 3D Seismic Attributes, Mathematics in Industry, Mathematical Methods and Modelling in Hydrocarbon Exploration and Production.* Springer, Berlin Heidelberg, 2006.

[12] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8:29–37, 1987.

[13] M. Livingstone. *Vision and Art: The Biology of Seeing.* Harry N. Abrams, 2002.

[14] E. R. Tufte. *The Visual Display of Quantitative Information.* Graphics Press, May 2001.

[15] J. W. Wallis, T. R. Miller, C. A. Lerner, and E. C. Kleerup. Three-dimensional display in nuclear medicine. *IEEE Transactions on Medical Imaging*, 8(4), 1989.

Chapter 5

Paper I: Illustrative Rendering of Seismic Data

Illustrative Rendering of Seismic Data

Daniel Patel[1,2], Christopher Giertsen[1], John Thurmond[3], Eduard Gröller[4,2]

Christian Michelsen Research, Bergen, Norway[1]
University of Bergen, Bergen, Norway[2]
Norsk Hydro, Bergen, Norway[3]
Vienna University of Technology, Austria[4]
Email: daniel@cmr.no, chrisgie@cmr.no
john.thurmond@hydro.com, groeller@cg.tuwien.ac.at

Abstract

In our work we present techniques for illustrative rendering of interpreted seismic volume data by adopting elements from geology book illustrations. We also introduce combined visualization techniques of interpreted and uninterpreted data for validation, comparison and interdisciplinary communication reasons. We introduce the concept of smooth transitions between these two semantical levels. To achieve this we present transfer functions that map seismic volume attributes to 2D textures that flow according to a deformation volume describing the buckling and discontinuities of the layers of the seismic data.

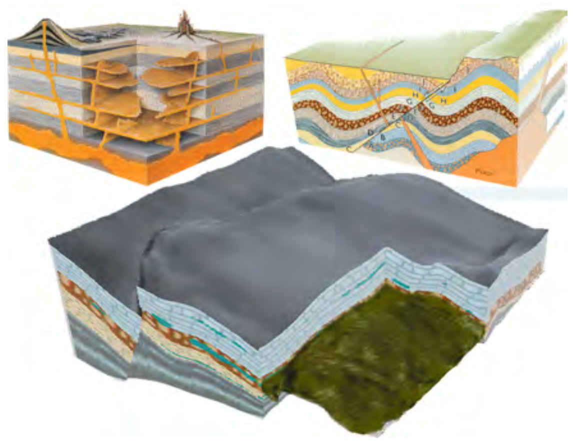


Figure 1: Geological and rendered illustrations. Top left: A cutout with extruding features. Top right: Textured layers with a fault discontinuity in the middle. Pictures are taken from Grotzinger et al. [6]. Bottom: Illustration rendered with our techniques.

1 Introduction

In geology faults and horizons are central subsurface structures. The earth has a layer-like structure and horizons are defined as the surfaces that separate one layer from another. Tension in the crust of the earth deforms the layers over time and creates cracks. These so called faults are more or less vertical discontinuities of the layers.

Geological illustrations in text books try to convey faults, horizons and other structures of the earth by using different artistic techniques as seen in the top of Figure 1. The illustrator draws a cubical subsection of the earth defining the area of interest. The horizons and faults are represented by using textures flowing inside the layers that are discontinuous across faults. The textures are drawn on the exterior side faces of the cubical subsection whose extent we hereby refer to as the roaming box. Axis-aligned cutouts with textures on the interior side faces are used to show features inside the cubical subsection. The cutouts sometimes contain extruding 3D features. Our illustrative renderings adopt all these techniques as seen in the bottom of Figure 1.

Figure 2 presents the flow from data acquisition to data visualization. The faults, horizons and other subsurface structures are discovered by geoscientists interpreting volumetric descriptions of the subsurface. These volumetric descriptions are typically obtained in geophysical surveys by processing the reflections of waves sent into the surface. The volume storing the reflection data is called the reflection volume. In a time consuming process the faults and horizons are manually found from the reflection volume and stored as surfaces. Several seismic attributes can be computed from the reflection data such as acoustic impedance (Ai) and the ratio between the pressure and shear wave (Vp/Vs). We will refer to these volumes as seismic attributes.

Coming up with a good visualization of interpreted data can be difficult, therefore we propose to use illustrative techniques. Illustrations are being used when there are certain high level aspects of a complex image, such as interpreted information, that need to be communicated in a simple way. Rendering of interpreted seismic data as illustrations has several advantages. It simplifies the visualization and emphasizes the elements of interest in order to disseminate gained knowledge from the interpretation process. Making a good illustration for scientific purposes takes time. Being able to render geological illustrations is advantageous both for quickly creating static images to illustrate geological books and for interactive oil exploration when interpreted survey data needs to be communicated as part of decision making.

Interpreting seismic data is a time consuming manual process and it is important to verify the interpretation with the underlying data source. By combining visualizations of interpreted and uninterpreted data it is possible to perform comparisons and look for deviations. This is another goal in our work. We propose to visualize the interpreted data as geological illustrations and to visualize uninterpreted data using color coded cutting planes and regular volume rendering. We present how to combine these two representations. The user can control the balance between these two visualization styles to fit his or her needs. For interdisciplinary communication reasons visualizations can be made to have the right balance between interpreted data which contains semantical information understandable by lay men to uninterpreted data which contains the information-rich underlying data material understandable by domain experts.

To our knowledge the concept of creating automatic illustrations of seismic data has not been thoroughly explored before, neither in the geophysics nor in the visualization research community. We also believe this applies to combined rendering of interpreted and uninterpreted seismic data.

We start with related work in Chapter 2. After an overview in Chapter 3 we describe the calculation of the texture flow in chapter 4. In chapter 5 we use the calculated flow in combination with texture transfer functions to texturize the cutting planes on the side faces of the cubical subsection and on the cutout. In chapter 6 we describe volume rendering for displaying the cutout and the surroundings and we specify how this is integrated with the rendering of textures during ray casting. Finally future work and conclusions are presented in chapter 7. The bottom half of Figure 2 shows a high level overview of the paper.

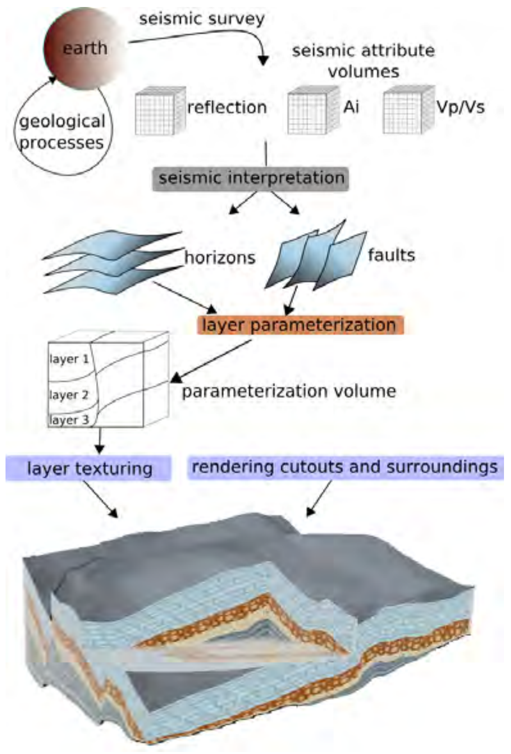


Figure 2: Overview of the process from data collection to visualization. The paper covers the lower three colored rectangles in chapter 4, 5 and 6.

2 Related work

We first review work dealing with illustrative techniques and then review work in the field of seismic visualization. Illustrative rendering is a non-photo realistic visualization technique using the advantages of conveying information through illustrations. In recent years several illustrative rendering techniques, mainly in the domain of anatomical visualization, but none in the domain of seismic visualization, have been proposed. Some of these techniques deal with applying textures from reference images.

Owada et al. [11] present an interactive system for texturing arbitrary cuts through polygonal objects. The user defines the texture flow by specifying a flow field and a distance field on the cut which is used in the texture synthesis to create a texture on the cut that follows the flow. Their method is general and therefore requires user interaction to specify the deformation and the texture. We calculate a parameterization up front so texturing can be achieved quickly and without the need for texture synthesis. In our approach many of the parameters defining the visualization are known prior to rendering, therefore less user specification is required

There are also several papers dealing with textures in medical volume rendering. Lu and Ebert [9] generate illustrative renderings of color and scalar 3D volumes by applying textures sampled from illustrations and photographs. 3D textures are created by combining color information from the illustrations with 3D volume data of a corresponding area. Finally the 3D textures are made tileable with Wang Cubes. With segmented volume data they apply the corresponding 3D textures on each segment. With unsegmented scalar data they use a transfer function to map scalar voxel values to the 3D textures in a similar way to what we propose. They do not deal with multi-attribute texture transfer functions and with deforming the textures to follow the underlying flow of the data as we do. In addition their method of calculating the textures is tailored to handle 3D textures whereas we use 2D textures.

Dong and Clapworthy [4] present a technique that achieves 3D texture synthesis following the texture orientation of 3D muscle data. Their algorithm has two steps. First they determine the texture orientation by looking at the gradient data of the volume and by using a direction limited Hough transform. Second they perform a volumetric texture synthesis based on the orientation data. In our work, instead of considering the volume for evaluating texture flow, we consider the geometric layers. In addition the texture synthesis of Dong and Clapworthy has the drawback of not working on textures with large interior variation as textures in geologic illustrations commonly have.

Wang and Mueller [14] use 3D texture synthesis to achieve sub-resolution zooming into volumetric data. With 2D images of several zoom levels of a tissue, they synthesize 3D volume textures for each level and use constrained texture synthesis during zooming to blend smoothly between the levels. They address the issue of sub-resolution details but do not consider texture flow.

In the domain of seismic processing GeoChron [10] is a formal model for parameterizing the layers defined by faults and horizons. The GeoChron model allows for several inputs which act as constraints to the parameterization. It considers the physical processes behind the deformation whereas our parameterization is fully defined by the fault and the horizon data. We believe that for illustration purposes a less physically accurate and computationally less intensive algorithm requiring a minimal amount of input and expertise such as our parameterization is preferable. However since our visualization algorithm is decoupled from the parameterization, it would also accept a GeoChron parameterization.

Cutouts on seismic data and interaction in VR was presented in the work by Ropinski et al. [13] where they use volume rendering with two transfer functions. One transfer function is used for the volume inside a user defined box and one transfer function is used for the volume outside. We incorporate and extend this concept in our work. Several papers on visualizing seismic data exist. Some deal with automatic horizon extraction [2] or fault extraction [2, 7], others deal with handling large volumes [2, 12], but none deal with illustrative rendering. Somewhat related is the dissertation of Frank [5] where the GeoChron parameterization [10] is used as a lookup to unfold and flatten a seismic data volume. In commercial systems seismic attribute data is presented with volume rendering and geometric surfaces are used to present horizons and faults.

3 Overview of the rendering process

Our methods render interactively the illustrative features found in geological images. Texturing is achieved by rendering deformed 2D textures on the exterior side faces of the roaming box and on the interior side faces of the cutout. For each layer the user assigns a texture and the texture's horizontal and vertical repeat rate. To also represent seismic attributes the user can assign textures and opacities to intervals of the seismic attribute values. These attribute textures are then blended and laid over the layer textures. We represent extruding features in the cutouts by volume rendering using a color transfer function together with a depth based opacity transfer function. The opacity is a function of the layer depth and the transparency can be set to restrict volume rendering to certain layers or to certain depths within a layer. Also in the surrounding area outside the cutout we perform volume rendering that can be restricted to certain layers or to certain depths within a layer. In the surrounding area the voxel colors are equal to the average color of the 2D texture used in the layer the voxel is in. This gives a consistent coloring with the cutting plane textures as can be seen in the bottom of Figure 2 and the top of Figure 10. There we render opaquely the top and bottom horizon with the average color of the 2D texture in the horizons. To visualize the uninterpreted seismic data we render the cutting planes and the surrounding volume with the color transfer function used for the cutout volume rendering. The user can smoothly change between the uninterpreted data rendering and the interpreted illustrative textured rendering by changing the blending factor. An overview of the texturing process can be seen in Figure 6 while the lower part of Figure 2 shows how the texturing fits into the 3D visualization. In the next three chapters the details of the visualization process described above is presented.

4 Layer parameterization

We parameterize the volume to render 2D planar textures following the flow of the layers and to achieve depth controlled volume rendering. Using the coordinate system shown in Figure 3d we define horizons as non-intersecting surfaces stacked in the z-direction of the type $z = H(x, y)$ and faults as non intersecting surfaces stacked in the x-direction of the type $x = F(y, z)$ or in the y-direction of the type $y = F(x, z)$. The faults, horizons and the side faces of the roaming box divide the volume into subvolumes which we will refer to as slabs. Conversely, each of these slabs is horizontally confined by what we will refer to as the upper and lower horizon and are laterally confined by fault surfaces and the side faces of the roaming box (see Figure 3a).

There exists no unique solution to parameterize a volume. We have designed the parameterization so that it represents the slabs in a flattened version where horizons and the layer between are planar. Figure 3d shows the parameterization coordinate system (u, v, w) embedded in the world coordinate system (x, y, z).

The parameterization consists of several steps. First the upper and lower horizon of the slab is extended by extrapolation (see dotted lines in Figure 3a). We do this extension to get a correct volumetric parameterization close to the vertical slab borders. Then the lower horizon surface is parameterized and the depth parameter w is calculated for the volume, (see curves in Figure 3b). Finally the 2D parameterization of the lower horizon is projected into the volume along the gradient field of the w parameter (blue curves in Figure 3c), resulting in a 3D parameterized slab.

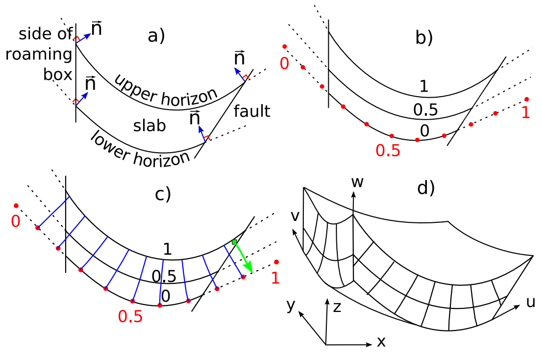


Figure 3: A 2D version of the steps needed for parameterizing a slab is shown in a-c. The world and the parameter coordinate system is shown in d.

Let $w_p = \{x, y, z\} \in \mathbf{R}^3$ be a point in W(orld space), and $p_p = \{u, v, w\} \in \mathbf{R}^3$ be the corresponding point in P(arameter space). We represent the mapping from W to P as :

$$\mathbf{P} : W \to P : p_p = \mathbf{P}(w_p) = \{P_u(w_p), P_v(w_p), P_w(w_p)\}$$

Let $min_{upper}(w_p)$ and $min_{lower}(w_p)$ express the Euclidean distance from w_p to the closest point on the upper and lower horizon respectively. The w parameter, or layer depth, is defined as:

$$P_w(w_p) = \frac{min_{lower}(w_p)}{min_{lower}(w_p) + min_{upper}(w_p)}$$

min_{upper} and min_{lower} are found by discretizing the upper/lower horizon into a point cloud. For discretizing we linearly subsample the horizon grid four times, and store the points in a kd-tree for efficient searching. Note that P_w does not express the distance to the closest surface as found by a distance transform, but the relative distance between the upper and lower horizon, it maps the lower horizon to 0, the upper horizon to 1 and is linear in between. In effect it flattens the layer and defines a local depth measure on it. See curves in Figure 3b.

We now have a w parameterization of the slab. The (u, v) values in the slab are found by projecting the (u, v) values from the parameterized lower horizon, which is described in 4.2. Projections into the volume is done along the streamlines seen as blue curves in Figure 3c which are defined by the vector field ∇P_w. ∇P_w is calculated using central differences. For each voxel we trace along the streamline in the opposite gradient direction toward the lower horizon, seen as a green arrow in Figure 3c. We assign to the voxel the (u, v) value of the intersected point on the lower horizon.

Assigning (u, v) values for the voxels inside the slab that are close to the vertical slab borders might result in streamlines leaving the slab and entering an area where P_w has not been calculated. See green arrow in Figure 3c. We have extended the horizons with the method described in 4.1 prior to the w parameterization and prior to the (u, v) parameterization of the bottom horizon. By doing this we have gradient data outside the slab as well as a parameterized surface outside the lower horizon which makes it possible to calculate streamlines leaving the slab. The parameterization procedure ensures that the (u, v) parameterization is orthogonal to the w parameterization which in turn will result in angle preservation in the 2D textures. The parameterization works well for surfaces of low curvature and without folds as seen in this application but would require some extension for handling other types of surfaces.

The parameterization is done on each slab and is stored in an RGB volume consisting of the (u, v, w) parameters. The parameters of each slab are all in the $[0, 1]$ range. To encode segmentation information for each slab we scale and shift the w and u parameter values. Each layer's w values are scaled and shifted such that values go from 0 at the lower horizon in the bottom layer to 1 at the upper horizon in the top layer with each layer having equally sized intervals. Similarly, the u values are scaled and shifted on each side of the fault. At the left side of the fault in Figure 4 the u values are between 0 and 0.5 and on the right side they are between 0.5 and 1. The segmentation information is used for having different textures in different layers and possibly on different sides of faults. The parameterization is not meant to be geologically accurate but to act as a tool for 2D texturing and depth dependent volume rendering. The goal is to achieve images with illustrative quality. The two following sections will describe the horizon extrapolation and the bottom horizon parameterization which was assumed to be done prior to the layer parameterization but were not explained in detail.

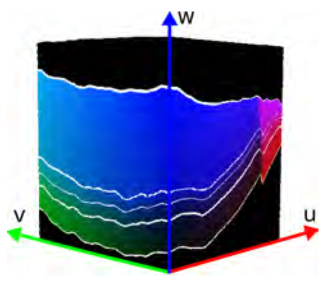


Figure 4: The parameterization RGB volume. White lines have been added on horizons. There is a color change across the fault due to shifting and scaling of the u parameter.

4.1 Horizon extrapolation

For parameterization of areas close to the vertical slab borders we need surface information beyond the horizon borders as described earlier. To achieve this we carry out a simple surface extrapolation in all directions. First we extrapolate the surface in positive and negative x direction by considering the surface as a collection of curves parallel to the x axis and extending the endpoints of the curves in tangential direction. See normals and dotted lines in Figure 3a. We then do the same procedure on the resulting surface in positive and negative y directions. Finally we crop the horizons so that their projections to the xy plane are rectangular and so that sufficient data exists beyond their original borders. On our data we ended up with a heuristic extension of 20 percent of the horizon length in each direction to correctly parameterize the areas close to the vertical slab borders.

4.2 Surface parameterization

For the (u, v) parameterization of the lower horizon surface we calculate a parameterization that locally minimizes the area distortion. Red dots on Figure 3b show the corresponding 1D version. The parameterization is created with the CGAL library [1] using the discrete authalic parameterization [3].

The parameterization defines (u, v) values for each vertex on the surface. The parameterization is constrained by giving initial values to the surface borders whose projection to the xy plane forms a rectangle due to the surface extrapolation. For the initial values we clockwise assign the border vertices with values (0,0) (0,1), (1,0) and (1,1) and interpolate the values along each edge with equidistant spacing. We now have a (u, v) parameterization of the lower horizon and a w parameterization of the slab.

4.3 Interpolation problem around horizons and faults

The parameterization volume is a discrete specification of our parameterization function. With trilinear sampling we get a smoother function which however leads to invalid interpolation in cells on slab boundaries where the eight cell corners are in different slabs. We calculate new values for the invalid corners by extrapolating from valid neighbor values. Then we perform the trilinear interpolation for the new corner values on the GPU. The extrapolation will try to assign a new value for invalid corners by considering the corner's two neighbors in positive x direction. If both are valid then their values are linearly extrapolated and assigned to the corner. If not, then the search continues in negative x direction and then similarly in y and z direction. In rare occasions the procedure fails to extrapolate all the invalid corner values and an erroneous interpolation is performed. The resulting artifacts will be noticeably only at the slab borders and will be of the same size as a voxel in the parameterization volume. The procedure improves the quality of the renderings as can be seen in Figure 5.

4.4 2D texture mapping on axis-aligned cutting planes

Our parameterization volume now makes it possible to apply an undeformed 3D texture stored in parameter space and deform it into world space for texturing voxels in our layers. However this would require to first generate 3D textures which is a research topic in its own as investigated by Lu and Ebert [9]. Since we are going to texture axis-aligned cutting planes as done in geological illustrations we can reduce the problem to a 2D texturing problem. This has several advantages. 2D tileable textures are easy to generate, take little space, and

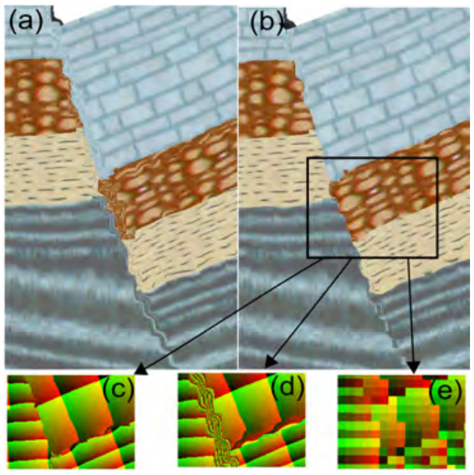


Figure 5: Fault and interpolation problems. In a) linear interpolation is used. In b) extrapolation as described in 4.3 improves the quality. In c) we see the parameterization of the zoomed-in rectangle with extrapolation as opposed to without in (d). In (e) we see the parameterization with nearest neighbor interpolation showing the resolution of the parameterization.

can be sampled from illustrations directly. With our method the 2D textures maintain coherency when moving the cutting planes and we have better control over the repetitive appearance than for 3D textures. However we need to define a transformation from 3D parameter space (u, v, w) to 2D parameter space (u', v'). The mapping is straightforward. For texturing in the xz plane we use the (u, w) values, for texturing in the yz plane we use (v, w) and for texturing in the xy plane we use (u, v) values. The mapping conserves the angle preservation property of the 3D parameterization.

5 Layer texturing

This chapter presents three transfer functions that are being used together to texture and color cutting planes. First we present the layer texture transfer function, abbreviated as layer TTF. It assigns textures to each layer. Then we present the scalar texture transfer function, abbreviated as scalar TTF. It assigns textures and opacities to regions having seismic attribute values in certain ranges. The resulting scalar TTF texture for a cutting plane is blended according to its opacities with the layer TTF texture using the over operator. The combined results are cutting planes with deformed textures similar to the ones in geology illustrations. Finally we present the concept of smoothly moving from illustratively rendered cutting planes to color coded cutting planes. Here seismic attribute values are mapped to colors using a color transfer function, abbreviated as color TF. See Figure 6 for an overview and Figure 7 for a texture example. Visualizing horizon, fault, deformation and seismic

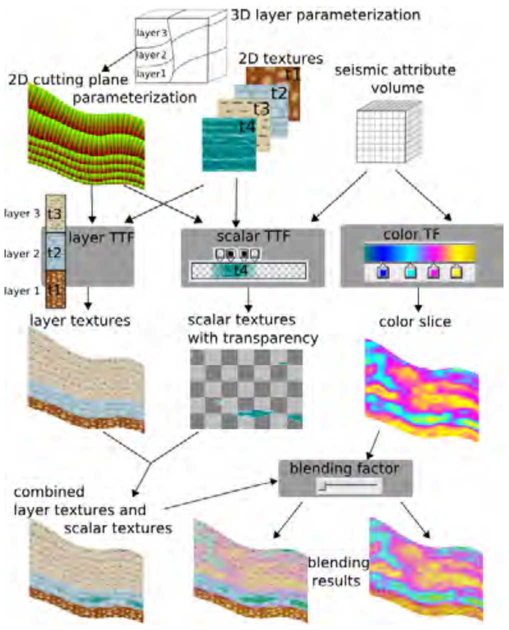


Figure 6: Overview of how textures are combined. Layer TTF, scalar TTF and color TFs are explained in 5.1, 5.2 and 5.3 respectively.

attribute information through textures has several advantages. By looking at Figure 7 one sees that textures communicate the id, orientation and compression of layers on a local scale. An example is given with the two small patches in the black circles in Figure 7. The texture of a patch reveals its layer id. The angles in the texture express the orientation of the layer in that area. Compression is presented through the vertical texture repeats in a layer. Since the vertical texture repeats are constant throughout the layer (there are always 16.5 bricks stacked in the height in layer 1 in Figure 7), compression will be high where the layer is thin and low where the layer is thick. It is possible to see that the texture patch in the left circle is slightly more compressed than the texture patch in the right circle of Figure 7. Finally, by letting both the horizontal and vertical texture repeat rate be a function of an underlying scalar value, scalar data can be presented in the texture as seen Figure 8. All this information is communicated with textures even on zoom scales where no horizons are visible, and also when zooming beyond the resolution of the seismic attribute volume. On such sub-resolution scales color transfer functions yield blocky or monotonous colored results whereas textures give aesthetically pleasing results. One can imagine zooming past the attribute volume resolution when inspecting overlaid high resolution data, such as bore well core data.

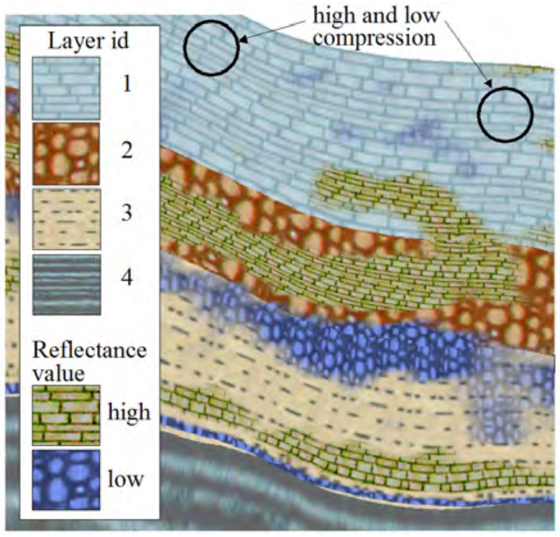


Figure 7: Combination of layer TTF and scalar TTF for the reflectance volume. The brown brick texture shows areas of high scalar values and the violet texture shows areas of low scalar values.

5.1 Layer texture transfer function (TTF)

To texture the cutting planes we use 2D RGB textures with wrap-around and bilinear filtering. They are taken from geological illustrations, and mapped on the layers. We use a layer TTF that maps from a voxel's w parameter value to a texture id, a horizontal and vertical texture repeat rate and an opacity value: $layer_{ttf}(w) = \{layerId, h_{rep}, v_{rep}, \alpha\}$. The opacity value is only used later in the cutout volume rendering. Since w varies in distinct intervals for each layer, each layer can have its own texture assigned. Texture variations within one layer such as having different textures in the top and bottom half of the same layer or having different textures on each side of a fault is also possible.

5.2 Scalar texture transfer function (TTF)

While the layer TTF represents the interpreted horizons as textures, the scalar TTF represents uninterpreted seismic attribute data as textures. The scalar TTF is equal to the layer TTF except that it is the seismic attribute values that are used as look up values. This makes it possible to control the textural appearance for regions on the cutting plane which have seismic attribute values in certain ranges. The scalar TTF texture is overlaid on the layer TTF texture using the over operator. The α value defines its transparency in the various regions. This combined view expresses how individual seismic attributes relate to layers, i.e., if intervals of an attribute are confined within certain layers or change significantly (or subtly) between layers. It also represents a unified visualization of layer data and seismic attribute data through textures.

Typically the repeat rates of a scalar texture are taken from the layer it is drawn on. However the user can set multiplicative factors in the repeat values in the scalar TTF to change this. We do this by default so textures can maintain the same repeat factors when crossing layers of different thicknesses. For a layer twice as thick as another one the vertical repeat of the thick layer's texture will be half of the thin layer's. A scalar texture crossing the layers would abruptly change its repeat rates. To have consistent repeats across layers as can be seen for the brown brick texture in Figure 7, the user can change in the layer TTF the vertical repeat for the thin layer to half of what it is for the thick layer.

The selection of repeat rates for the textures is highly dependent on the degree of zoom. When zooming out, textures will be perceived as being too high frequent and when zooming in they will be perceived as being too low frequent. For this reason we multiply all the repeat factors with a global user definable repeat factor which is manually set according to the zoom level.

5.3 Rendering uninterpreted and interpreted data

To inspect the uninterpreted data directly on cutting planes we apply the color TF on the scalar values of a seismic attribute. We also introduce the concept of a continuous transition from illustrative rendering of interpreted data to rendering of uninterpreted data. The transition is done by smoothly blending from visualizing textured cutting planes to visualizing cutting planes colored by the color TF with seismic attribute values. This not only gives a smooth transition from one mode to the other but also introduces an intermediate rendering mode where interpreted data is superimposed on the uninterpreted data. The balance between the two data sources can be adjusted to get what the user perceives as an optimal balance between the rendering techniques. See Figure 10.

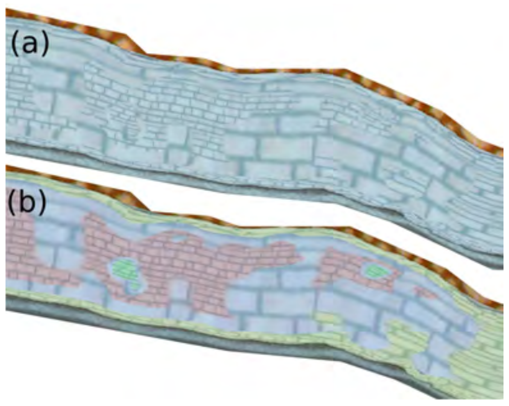


Figure 8: Layer TTF, scalar TTF and color TF combined. Instead of using different textures on intervals of the scalar values we use the same texture with four different repeat rates. It is difficult to discern the textures in a). In b) we blend in colors from the color TF to more easily discern the textures.

6 Rendering cutouts and surroundings

We implement volume rendering with one transfer function for the cutout and another one for the surroundings to support different rendering styles. By doing this we can achieve rendering of extruding features in the cutout and opaque ground rendering in the surroundings as seen in geological illustrations.

For volume rendering in the cutout we use the color TF on seismic attribute data introduced earlier. To specify transparencies in the volume rendering we extend the color TF with an α channel. By multiplying a voxel's α value from the color TF with the α value from the layer TTF we can adjust the transparencies based on the w value of the sample. Now we can do volume rendering on selected layers by manipulating the α in the layer TTF and making layers transparent or semitransparent.

For visualizing the surroundings we do volume rendering where each voxel is given the average color of the 2D texture at the voxel position. The average color is precalculated for each 2D texture. The opacity is controlled by a separate opacity transfer function for the surroundings. It maps the w parameter to opacities enabling a layer oriented volume rendering of the surroundings. The opacity can then be set for instance to render certain horizon surfaces or layers semitransparently. When performing smooth transitions from rendering of interpreted data to rendering of uninterpreted data we go from using the average color of the 2D texture at the voxel position to using the voxel's color according to the color TF and the seismic attribute value at that position. In the images of this article we render the top and bottom horizon opaquely to get an opaque ground as seen in geological illustrations.

In the following paragraphs we describe how volume rendering is combined with texturing of the cutting planes. We perform ray casting with empty space skipping as suggested by Krüger and Westermann [8]. The entry and exit point of each ray is further clipped to the roaming box. Volume rendering with the transfer function for the surrounding is performed outside the cutout, while the transfer function for the cutout is used inside the cutout. Texturing is done at points where the parameterization volume intersects the exterior of the roaming box or the interior of the cutout box. See Figure 9 for a 2D depiction which acts as a reference to the following description. Texturing is done at the entry

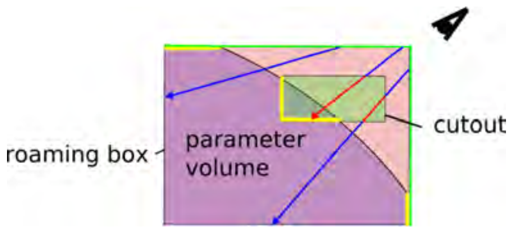


Figure 9: Combining volume rendering with texture rendering. The green line depicts the entry points. The yellow lines show where texturing is done. Red ray segments show where the cutout transfer function is used and blue ray segments show where the surrounding transfer function is used

point if the entry point is inside the parameter volume (green/yellow border). If not volume rendering with the transfer function for the surrounding is performed until the end point (upper blue ray) or until the cutout is intersected. If the cutout is intersected then volume rendering with the transfer function for the cutout is used (red segments) until the cutout exit point is reached. If the cutout exit point is inside the parameter volume texturing is performed (yellow border). If not, ray casting with the transfer function for the surrounding is performed until the exit point. A ray is always terminated if opacity reaches 1.

By doing volume rendering only on selected layers we can easily achieve the effect seen in geological illustrations of extruding layers in the cutouts. For exploration of the seismic data this is useful when the user wants to consider only one layer at a time. For instance the oil reserves are typically trapped between horizons in so called reservoirs. If the expert wants to perform volume rendering to explore such a reservoir it would be natural to confine the volume rendering to the layer the reservoir is in. See the bottom of Figure 1 for an example of volume rendering in a cutout limited to a layer. It shows a combined texture and volume rendering with an extruding layer. Volume rendering is performed only for layer 3 with brown color to mimic a geological illustration. The layer discontinuity is due to a fault. Turquoise patches on the textures show areas with high reflection values. Figure 10 shows a smooth transition from illustrative rendering to seismic attribute rendering.

The texture calculation and volume rendering is performed on the GPU in a single pass. With a Geforce 8800 GTX graphics card and an image size of 800×800 we achieve 5 frames per second. Without the extrapolation as described in 4.3 the frame rate is doubled. The three component parameterization volume is of size $128 \times 128 \times 128$ and the reflectance volume of size $240 \times 271 \times 500$. The Ai volume is of size $96 \times 96 \times 500$ and covers a smaller area than the reflectance volume. The 2D textures are each of size 64×64.

7 Conclusions and future work

We have presented a technique for illustrative rendering of interpreted geological data. We have also shown how to create combined visualizations of interpreted and uninterpreted seismic data for validation and comparison reasons and for creating visu-

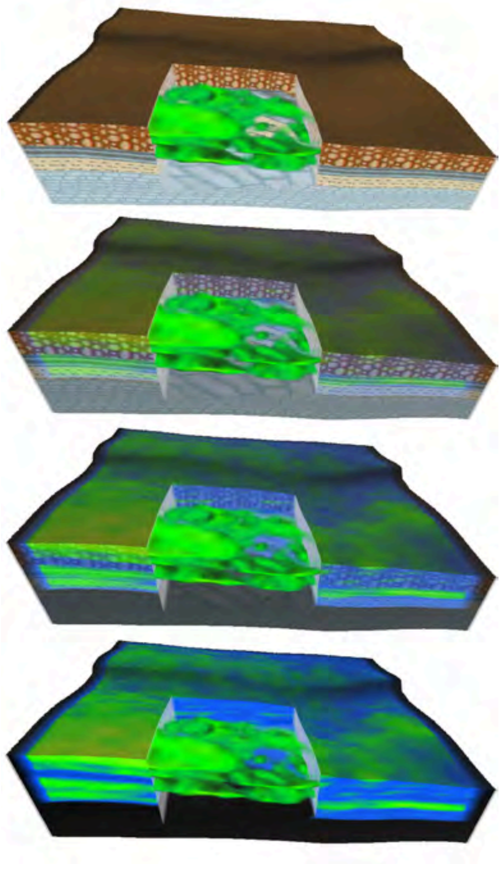


Figure 10: Blending from illustrative rendering to uninterpreted data rendering of the Ai attribute. Volume rendering is performed in areas with high Ai values. On the right of the cutout one can see how the green area having high Ai values corresponds to a layer. The black areas contain no Ai data.

alizations that can be targeted to anyone from laymen to domain experts. On the technical side we have presented the concept of 2D texture transfer functions with deformed textures.

Illustrative techniques can make it faster to evaluate large oil prospects. It can also improve communication between different stakeholders and towards media, public sector and politicians. In the future we will look into methods making it possible to do illustrative rendering of uninterpreted data.

References

[1] CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

[2] L. Castanie, B. Levy, and F. Bosquet. Volume-explorer: Roaming large volumes to couple visualization and data processing for oil and gas exploration. *Proceedings of IEEE Visualization '05*, pages 247–254, 2005.

[3] M. Desbrun, M. Meyer, and P. Alliez. Intrinsic parameterizations of surface meshes. *Computer Graphics Forum*, 21:209–218, 2002. Eurographics conference proceedings.

[4] F. Dong and G. Clapworthy. Volumetric texture synthesis for non-photorealistic volume rendering of medical data. *The Visual Computer*, 21(7):463–473, 2005.

[5] T. Frank. *Advanced Visualisation and Modeling of Tetrahedral Meshes*. PhD in geosciences, Institut National Polytechnique de Lorraine, 2006.

[6] J. Grotzinger, T. H. Jordan, F. Press, and R. Siever. *Understanding Earth*. W. H. Freeman and Company, 1994.

[7] W.-K. Jeong, R. Whitaker, and M. Dobin. Interactive 3d seismic fault detection on the graphics hardware. *Volume Graphics*, pages 111–118, 2006.

[8] J. Krüger and R. Westermann. Acceleration techniques for gpu-based volume rendering. *Proceedings of IEEE Visualization '03*, pages 38–47, 2003.

[9] A. Lu and D. S. Ebert. Example-based volume illustrations. *Proceedings of IEEE Visualization '05*, pages 83–92, 2005.

[10] R. Moyen. *Parametrisation 3d de lespace en geologie sedimentaire: le modele geochron*. PhD in geosciences, Institut National Polytechnique de Lorraine, 2005.

[11] S. Owada, F. Nielsen, M. Okabe, and T. Igarashi. Volumetric illustration: designing 3d models with internal textures. *SIGGRAPH '04*, pages 322–328, 2004.

[12] J. Plate, M. Tirtasana, R. Carmona, and B. Fröhlich. Octreemizer: a hierarchical approach for interactive roaming through very large volumes. *Proceedings of VISSYM '02*, pages 53–64, 2002.

[13] T. Ropinski, F. Steinicke, and K. H. Hinrichs. Visual exploration of seismic volume datasets. *Journal Proceedings of WSCG '06*, 14:73–80, 2006.

[14] L. Wang and K. Mueller. Generating sub-resolution detail in images and volumes using constrained texture synthesis. *Proceedings of IEEE Visualization '04*, pages 75–82, 2004.

Chapter 6

Paper II: The Seismic Analyzer: Interpreting and Illustrating 2D Seismic Data

The Seismic Analyzer: Interpreting and Illustrating 2D Seismic Data

Daniel Patel, Christopher Giertsen, John Thurmond, John Gjelberg, and M. Eduard Gröller, *Member, IEEE*

Abstract—We present a toolbox for quickly interpreting and illustrating 2D slices of seismic volumetric reflection data. Searching for oil and gas involves creating a structural overview of seismic reflection data to identify hydrocarbon reservoirs. We improve the search of seismic structures by precalculating the horizon structures of the seismic data prior to interpretation. We improve the annotation of seismic structures by applying novel illustrative rendering algorithms tailored to seismic data, such as deformed texturing and line and texture transfer functions. The illustrative rendering results in multi-attribute and scale invariant visualizations where features are represented clearly in both highly zoomed in and zoomed out views. Thumbnail views in combination with interactive appearance control allows for a quick overview of the data before detailed interpretation takes place. These techniques help reduce the work of seismic illustrators and interpreters.

Index Terms—Seismic interpretation, Illustrative rendering, Seismic attributes, Top-down interpretation

1 Introduction

Oil and gas are valuable resources accounting for around 64% of the total world energy consumption [10]. Oil and gas search and recovery is an economically valuable but complex task. Imaging the subsurface for exploration purposes is highly expensive. The imaging surveys consist of sending sound waves into the earth and recording and processing the echoes. Throughout the article we refer to this processed data as the seismic reflection data.

Due to the measuring expenses, an iterative approach for collecting data is taken. The first stage of a search typically involves collecting multiple 2D seismic slices which are analyzed by a team of geologists and geophysicists. If an area showing signs of hydrocarbons is discovered, 3D seismic reflection data is collected and analyzed. If further indications of hydrocarbon accumulation are found in the new data, drilling a well might be considered. Irrespective if the drilling hits a reservoir or not, it will give deeper insight into the data due to the process of bore hole logging. A bore hole log consists of physical measurements along the well path such as mineral conductivity, radioactivity and magnetism.

The current work flow in searching for oil and gas is to start a detailed interpretation of seismic structures. When enough structures have been interpreted to get an overview of the data, the results are discussed by an interdisciplinary team. This bottom-up approach is time consuming. The interpretation is challenging due to the low resolution and noisy nature of the seismic data. In cases of doubt, the interpreter often creates several alternatives of the same seismic structure. In addition, it is not uncommon that the team disagrees on the interpretation and decides that parts of the data must be reinterpreted. As soon as a consensus is achieved, the interpretation is documented for further dissemination outside the team. As part of the documentation, a seismic illustrator draws illustrations of the interpretation. Traditional illustrating is a time-intensive task and is therefore done late in the work flow.

- *Daniel Patel is with Christian Michelsen Research, Bergen, Norway, E-mail: daniel@cmr.no.*
- *Christopher Giertsen is with Christian Michelsen Research, Bergen, Norway, E-mail: chrisgie@cmr.no.*
- *John Thurmond is with StatoilHydro, Bergen, Norway, E-mail: jothu@StatoilHydro.com.*
- *John Gjelberg is with StatoilHydro, Bergen, Norway, E-mail: johgje@StatoilHydro.com.*
- *Eduard Gröller is with Vienna University of Technology, Austria, E-mail: groeller@cg.tuwien.ac.at.*

Manuscript received 31 March 2008; accepted 1 August 2008; posted online 19 October 2008; mailed on 13 October 2008.
For information on obtaining reprints of this article, please send e-mailto:tvcg@computer.org.

In this work we present a top-down interpretation approach as a step prior to the currently used bottom-up interpretation. This alleviates some of the issues in the bottom-up approach. The top-down interpretation is performed with a sketching tool that supports coarse interpretation and quick creation of communication-friendly seismic illustrations. The inaccuracy in the seismic data and the need for preciseness during bottom-up interpretation can easily lead the interpreter to wrong conclusions based on insufficient information. When reinterpretation is required due to wrong conclusions, the long time lapses between interpretation and meetings is a big problem. With our top-down procedure, we facilitate short interpretation-to-meeting cycles. This enables earlier and more frequent discussions of interpretation hypotheses. Hypotheses can be clearly annotated, presented and possibly discarded earlier. A common understanding of the data can be achieved before the detailed interpretation starts. This can reduce the need for reinterpretations during the bottom-up approach. One can save time identifying which structures to focus on during the bottom-up interpretation. Also, no tools for quickly creating seismic illustrations exist. Currently general drawing software is used to illustrate seismic slices.

We focus on interpreting and illustrating 2D slices instead of 3D volumes for several reasons. Only 2D data exist in the earliest stage of data collection. Also it is common to do a slice by slice interpretation of the seismic volume. Furthermore, when creating illustrations for communication purposes, illustrators make heavy use of 2D slices. Even for 3D illustrations, cutouts are often used with 2D illustrated slices on the cutout side surfaces as seen in Figure 1.

1.1 Interpreting Seismic Data

Oil and gas is created when organic material is deposited and then buried, followed by the application of pressure and heat over a long period of time. The produced oil and gas will migrate upwards and accumulate in reservoir structures such as anticlines (seen in left Figure 1) or fault traps. Oil and gas is searched for by looking for signs of these depositional or reservoir structures. Due to the limited resolution of seismic data, one must work with interpretable events such as the horizons. Horizons delineate rocks with different mineral densities or porosity characteristics. The horizons can be seen as bright or dark lines in gray-level reflection data.

The existing bottom-up work flow at the Norwegian oil company StatoilHydro is to interpret about each 25th yz slice along the x-axis and then about each 25th xz slice along the y-axis. For each slice, typically around 5 key horizons are identified and traced out. It takes about 2-3 hours to interpret one slice of a standard-sized dataset. Certain points at a horizon might have several continuation options. In these cases, each option is interpreted and stored as a separate horizon. Deciding which horizon is correct, and which is not, is taken later when a better overview exists. When all slices are interpreted, different methods of automatic growing and surface interpolation are performed to complete the horizons in the slices that were jumped over. After an in-

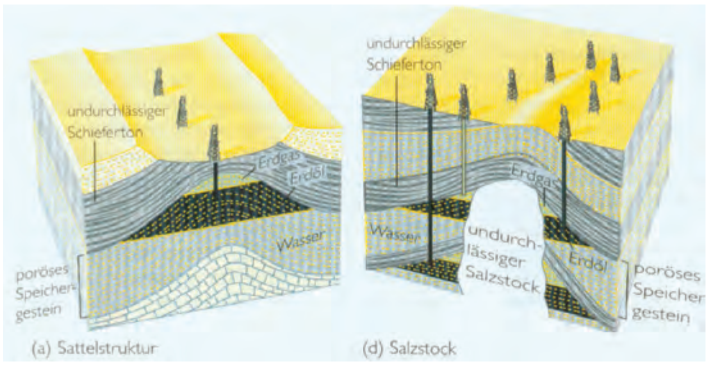


Fig. 1. Hand crafted illustrations of an oilfield. Left: Notice how the brick texture in the lowest layer and the stippled lines in the oil area are bending along the strata. Pictures are taken from Grotzinger et al. [8]

terpreter is satisfied with the horizons, a multidisciplinary discussion will take place to discuss the interpretation.

In addition to the horizons, the stratigraphy of the data is an important feature during interpretation. The term stratigraphy originates from the word stratum, meaning a layer of rock or soil with internally consistent characteristics that distinguishes it from contiguous layers. Strata can be found by either searching for areas with certain textures, or groups of horizons with common trends. This is referred to as seismic textures in literature [10]. Figure 2 gives some specific seismic examples. To support interpretation, different types of seismic textures in the seismic reflection data can be calculated automatically. Calculating the seismic texture *chaos* for a slice results in a new slice where the value of each pixel encodes the variance of the gradients in the pixel's neighbourhood (seen in bottom right Figure 3). The chaos attribute helps to indicate areas where horizons are difficult to track or to indicate stratigraphic regions containing salt (white area in right Figure 1). Such calculated slices are referred to as derived attributes. Other examples of derived attributes are reflection intensity and dip. Reflection intensity is the amplitude of the reflection strength in frequency space. Areas of strong reflections will have high reflection intensity. Dip is a measure of the angle of the seismic data in a neighbourhood by looking at the gradient values of the reflection data. See Figure 3 for some attribute examples.

In the domain of seismic interpretation, we facilitate visualization of derived attributes. We introduce the novel approach of identifying the horizons in the reflection data in a preprocessing step, and representing them as a collection of curves. We also present a filtering on the identified horizons so that they can be grouped together into strata according to similarity properties. Finally we enable the visual representation of well logs to be spread along horizons crossing the well.

1.2 Illustrating Seismic Data

We present novel illustrative visualization techniques of seismic data inspired by geological illustrations such as Figure 1. The patterns used in geological illustrations convey rich information. There exist over a hundred unique seismic patterns, each representing different types of rock-formation characteristics, also known as lithologies. The US Federal Geographic Data Committee [2] has defined a standard and a symbol lexicon for these patterns. The patterns are cleverly designed

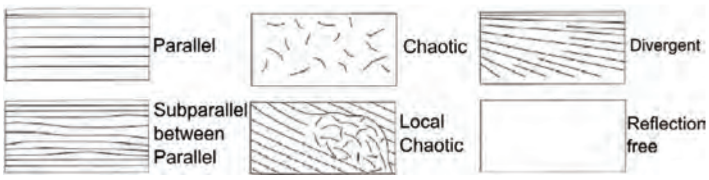


Fig. 2. Examples of stratigraphic patterns a seismic interpreter looks for during interpretation. Image is from Iske and Randen [10].

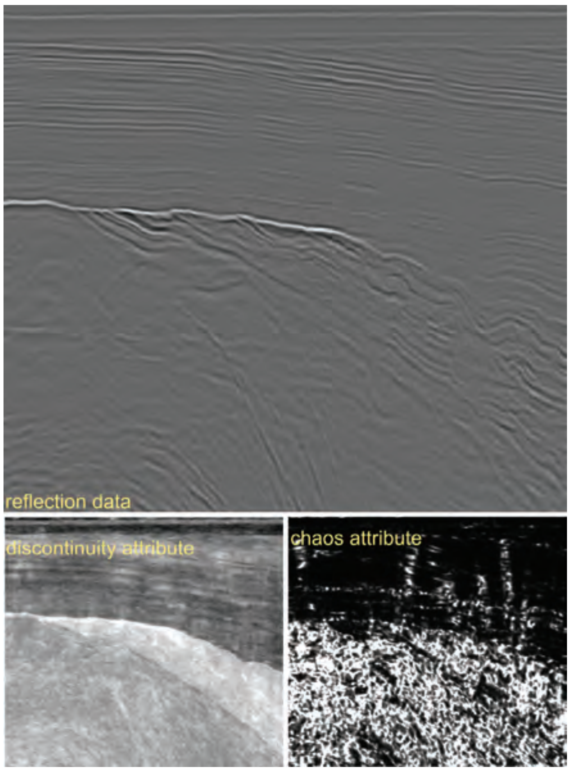


Fig. 3. Slices with reflection data at top and derived attributes below.

and constitute a taxonomy. Lithologies of the same subgroup are represented with similar patterns. Thus, even if a geologist does not recognize the exact lithological meaning of a pattern, he should be able to identify the general group it belongs to. Patterns or textures, and the way they bend are crucial information in geology. We explore the use of textures at early interpretation stages to easier express the interpreters inner model of the data.

The data modalities that drive our illustrative process are the derived attributes, the traced horizons, and the well logs. Our illustrative rendering techniques map these data modalities to textures and lines that bend along the horizons in the seismic reflection data. To achieve the 'bending' of textures and lines, we calculate a 2D parameterization of the seismic reflection data. The mappings from data modalities to illustrative renderings are defined in terms of texture and line transfer functions. Each mapping of a modality results in what we call an illustrative layer. The rendering order of the illustrative layers defines which layers are in the background as context and which layers are in the foreground as focus. Different illustrative mapping techniques are defined for each seismic modality. Horizon data is either manually selected by user-picking or filtered according to horizon properties such as angles or average reflection strength. The selected horizons are then either mapped to textures with the texture transfer function or rendered as coloured lines using the line transfer function. For well logs, textures can be assigned to intervals of the well log values and spread out to a user-defined width along the horizons that cross the well. Finally, derived attribute data can be mapped by assigning intervals of the values to either textures with opacities, or to coloured lines with opacities. These rendering algorithms make it possible to merge all the data modalities into one multi-attribute illustrative image. For an intuitive understanding of the results, the mappings have to be carefully chosen. These techniques result in informative images that can be used directly as illustrations. This reduces the illustrator's time consuming manual work for tasks such as drawing deformed textures. An overview of the seismic analyzer can be seen in Figure 4.

In the domain of seismic illustrative rendering, the novelty of our approach lies in the parameterization of the seismic slice which allows for texturing, for line drawing and for dragging out textures from well logs. Also we introduce the concept of mapping seismic modal-

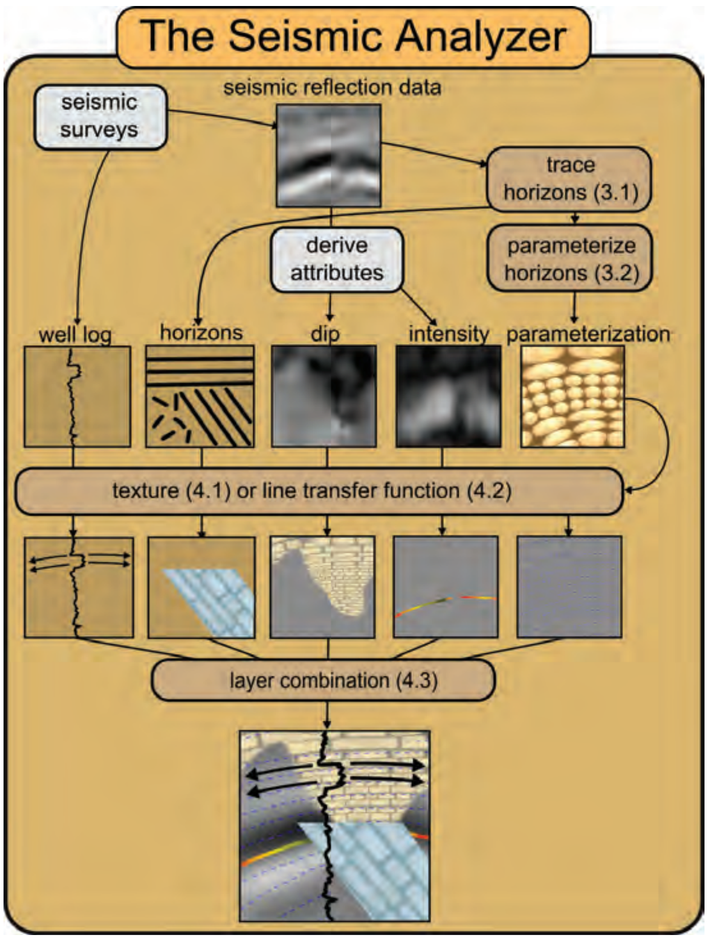


Fig. 4. The seismic analyzer. The brown rounded rectangles represent our algorithms and refer to the sections describing them. The 'seismic surveys' rectangle represents the process of obtaining the seismic data. The 'derive attributes' rectangle represents the process of deriving attributes using external software.

ities to illustrative layers which achieves multi-attribute visualization, illustrative visualization, and scale invariance.

2 Related Work

Several papers discuss processing and visualization algorithms for 3D seismic data. The visualization algorithms presented in papers and in commercial solutions are mostly direct volume rendering of the seismic data and surface rendering of interpreted objects such as horizons. In slice visualizations, horizons are represented as lines. For accurate structural interpretation, some papers deal with horizon extraction as in Castanie et al. [5] or fault extraction as in Jeong et al. [11] and Gibson et al. [7]. Pepper and Bejarano [15] give an overview of automatic interpretation methods. Plate et al. [16] and Castanie et al. [5] deal with handling large seismic volumes. Ropinski et al. [18] cover volume rendering of seismic data in VR. They present spherical and cubic cutouts which have a different transfer function than the surrounding volume. Commercial software used in oil companies include HydroVR [13] and Petrel [1]. None of these works deal with illustrative techniques or top-down interpretation as presented here. Papers that cover horizon extraction do it semi-automatically. The user has to select a seedpoint from where the horizon will be grown according to some user defined connectivity criteria. Our work differs in that we automatically pre-grow all horizons.

In our previous work [14] illustrative techniques for seismic data was also presented. That work dealt with the presentation and validation of *interpreted* seismic data. In this work we introduce a toolbox for interpreting and illustrating *non-interpreted* seismic data. This provides stronger illustrating capabilities than in our previous work. The parameterization in the previous work [14] required manually interpreted *complete* horizons. Now we introduce a parameterization that works directly on uninterpreted data by accepting automatically created horizon *patches*. We extend the texture transfer function from the previous work by introducing a flexible GUI and layered illustrative transfer functions for lines, wells and the automatically extracted horizons. Furthermore we propose how these techniques can be used in concert to improve the current work flow in oil companies by introducing the concept of top-down interpretation.

The 2D parameterization we present in this paper differs from parameterization methods used for texturing in other domains. 2D textures are applied to 2D images in both vector field visualization, such as Taponecco et al. [19], and in brush stroke synthesis as in Hays and Essa [9]. Our method differs in that we calculate the texture bending so it follows structures specific to seismic reflection data and not general gradient trends or edges. How we create the 2D parameterization is also different. Taponecco et al. [19] create 1D lines that are parameterized in length and then expanded in thickness to create a 2D parameterization. This procedure is performed locally over a collection of evenly distributed lines that cover the 2D image. This however results in overlapping textures. Our method considers the 2D space globally to create a complete and non-overlapping parameterization.

Much work has been done in multi-attribute visualization, Bürger et al. [4] present a state of the art overview. Crawfis and Allison [6] present a general framework where textures, bump maps and contour lines are used for multi-attribute visualization. Kirby et al. [12] present multi-attribute visualization of 2D flows using concepts from painting. They visualize flow attributes using procedural glyphs on a colour-coded background. Taylor [20] takes a general approach by combining colour, transparency, contour lines, textures and spot noise. He succeeds in visualizing four attributes simultaneously. However, little work has been done in multi-attribute visualization of seismic data.

3 Extracting Horizons and Parameterizing the Reflection Data

In this section we describe how the horizon structures are found and how the 2D parameterization is calculated. The horizon lines are used directly in the visualization of the data and as input to the 2D parameterization. The horizons and the parameterization is calculated in a preprocessing step prior to the visualization.

3.1 Tracing the Horizons

We adapt the method described in Iske and Randen [10] to trace out horizons. By considering the seismic reflection data as a height field, the horizons are running along valleys and ridges of the height field. We automatically trace out some of these valleys and ridges. Our method differs from existing horizon tracing algorithms since it does not require a user defined seed point for each trace. We go through all samples in the seismic slice and create traces for samples that are local maxima or minima in a vertical neighbourhood of 3 samples. The result is a collection of lines going through the horizons of the slice.

3.2 Parameterization of the Horizons

To achieve the effect of textures and lines following the orientation trend of the underlying reflection data, we create a parameterization from the traced horizons. The parameterization creates a relationship between image space and parameter space as seen in Figure 5a and b. There are four steps in determining the parameterization. The first step is to create suggestive line segments that indicate the horizons in the reflection data. We use the extracted horizon lines for this. The second step is to calculate the vertical v parameter values from the horizons. Thirdly, from the v parameter values, the horizontal u parameter values are calculated. Finally the u parameter values are normalized to minimize distortion in the parameterization. The parameterization process will ensure that horizons are mapped to straight lines in parameter space. Inversely, this guarantees that straight illustrative textures and

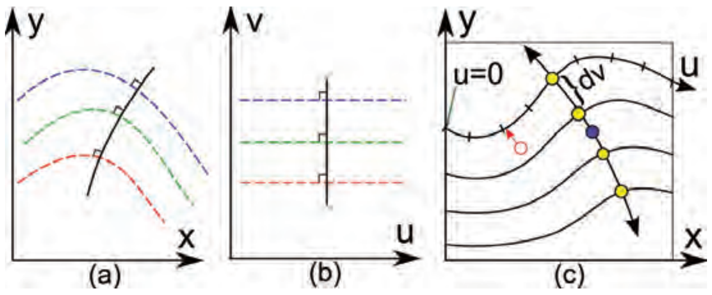


Fig. 5. Relationship between image space (a), and parameter space (b). In (c) is shown the procedure of finding the *u* parameter for a point (red circle) in image space by tracing from the point along a curve normal to the *v* parameter until it hits a *u* parameterized *v*-isocurve.

straight lines in parameter space will be aligned with the horizons in image space.

In the second step we calculate the *v* parameter by sweeping a vertical line from left to right over the horizons (Red line in Figure 6). The sweep line consists of a set of control points with unique *v* values. Initially there is one control point at the bottom of the line with *v*=0 and one at the top with *v*=1. The *v* values between control points are linearly interpolated. As the sweep line moves to the right, it will intersect the horizon lines. At the position on the sweep line where it intersects the start of a horizon, a new control point is created (double circle) which is assigned the interpolated *v* value at that point on the sweep line. For the following intersections of the same horizon line, the associated control point will update its position according to the intersection but will keep its initially given *v* value. One can imagine the sweep line as a rubber ribbon getting hooked on and off the horizons. After all points in the slice have been assigned *v* values, a 2D smoothing is performed to smooth out the discontinuities that arise just behind the horizon ends (see Figure).

Finding the *u* parameterization involves finding a mapping from vertical lines in parameter space (vertical black line in Figure 5b) to image space (Figure 5a). We want the parameterization to be angle preserving so that for instance the 90 degree angles at the edges of bricks in a brick texture are more or less preserved when mapped to image space. For this to be fulfilled, we require that the vertical lines in parameter space are always normal to the *v*-isocurves. We find an initial vertical line in parameter space by tracing two lines from the middle of the image space (blue dot in Figure 5c), one in the normal direction of the *v* parameterization, and one against the normal direction. This line is then parameterized according to its curve length and is divided into intervals of length dv (yellow dots in Figure 5c). From each of the interval ends (yellow dots) we span out *v*-isocurves. Each *v*-isocurve is *u*-parameterized according to its curve length and is set to 0 at the intersection with the left image border. The image space has now been divided into strips. Finally, the *u* value for any point in

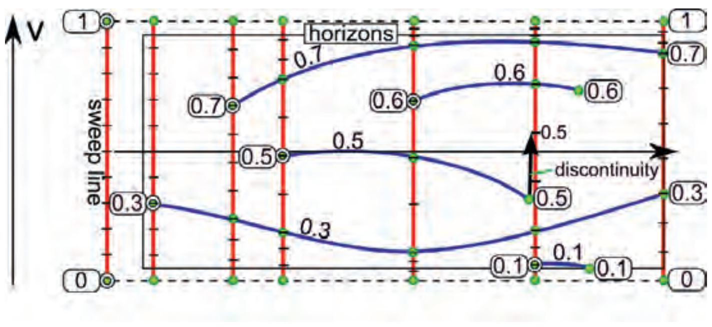


Fig. 6. Calculating the *v* parameterization by sweeping the red line with green control points from left to right. The blue lines are the horizons. The numbers are the *v* parameter values of the control points. Values in between the control points of a line are linearly interpolated.

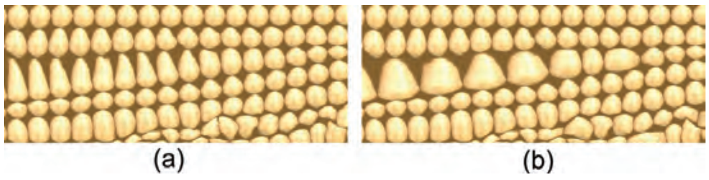


Fig. 7. Part of the reflection data textured with a ball texture to present the parameterization. Before (a) and after (b) isotropic correction.

a strip is found by tracing from the point's position in the *v* gradient direction until a *v*-isocurve is hit (red line in Figure 5c). The point's *u* value is set to the *u* value at the isocurve intersection.

We illustrate the resulting parameterization with a ball texture in Figure 7a. Each row of balls represents a parameter strip. One can see that the balls in the third row from the top become stretched and anisotropic as the strip's upper and lower *v*-isocurves diverge. We correct the parameterization so that the *u*/*v* ratio is constant as can be seen in the right image. This is done by remapping the *u* parameter for each strip so that it increments along the curve length relative to the thickness of the strip at that point. This ensures that textures are drawn with a consistent width/height ratio. The texture size however does vary. Varying texture sizes can be useful information during seismic interpretation since they communicate the degree of divergence of the horizons in an area.

4 Illustrative Rendering of Seismic Modalities

We present visualization techniques that filter and map the seismic modalities to visual representations which are more intuitive to understand than their direct representations. Combining filtering and mapping of the data to a visual representation has several advantages. Firstly, often only certain intervals in the value range of a modality are of interest to show. Uninteresting value ranges can be set to have no visual representation and value ranges of interest can be mapped to prominent visualizations. Secondly, scale invariance is achieved by changing the sparseness of the visual representation. By this we achieve that the image space is neither underloaded nor overloaded with visual information no matter how small or large the slice is. This will be discussed in more detail in section 5.1. Our approach also allows for multi-attribute and focus+context visualization where one can apply special rendering styles for modalities and value intervals that are of particular interest. In cases where the visual representations of several illustrative layers overlap, one can define a higher importance for one illustrative layer by rendering it on top of the others.

We have defined two generic techniques for mapping a seismic modality to an illustrative layer, i.e. texture transfer functions and line transfer functions. These two techniques define the data filtering and representation assignment for the seismic modalities. Some modalities have additional parameters that determine their appearance. By manipulating the texture and line transfer functions, the domain expert has a high degree of freedom to visualize and explore the multi-attribute data in real time. A texture transfer function assigns opacities and textures to a value range whereas a line transfer function assigns opacities and coloured lines to a value range. In effect the opacity assignment defines the data filtering, and the texture or colour assignment defines the representation mapping.

4.1 The Texture Transfer Function

The texture transfer function maps the scalars of a modality to an opacity and to a texture. The opacity is defined by a graph along the scalar axis. The scalar-to-texture mapping is described by discretely positioned texture references along the scalar axis. Scalars between two texture references will be represented by a weighted blend of the adjacent textures. The weighting is defined by the relative distance to the texture references. Examples of texture transfer functions defined in our GUI can be seen on the left side of Figure 8.

The textures are mapped to image space by a 2D parameterization. Either the parameterization of the reflection data or a basic uniform

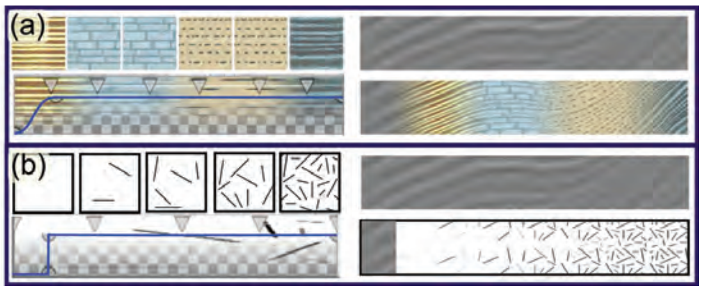


Fig. 8. Left: Texture transfer functions with blue graphs defining opacities. (a) Right: Textures following the parameterization of the reflection data. (b) Right: Textures following a uniform parameterization. Texture lookup values increase linearly from left to right in (a) and (b).

axis aligned parameterization which is independent of the horizon parameterization, is used. The latter parameterization is best suited for visualizing areas which have poorly defined horizons. Examples are areas that are chaotic, have weak or no reflections or that contain the geologic discontinuities called faults. Figure 8a and 8b show the two different types of parameterization. For each illustrative layer, multiplicative factors of the horizontal and vertical repeat rate of the textures must be defined by the user. We use the texture transfer function to control the visualization of the horizons, the well logs and the derived attribute slices.

For the traced horizon lines, we calculate measures like length, strength and angle. Each horizon line has a segmentation mask around it where the texturing will take place. The user can decide which horizon measure the transfer function will use. An example of a horizon transfer function is seen in Figure 9b. It is also possible for the user to pick, with the mouse, a subset of the horizons to apply the texture transfer function on.

Well logs, being physical measurements along a vertical line in a slice, are represented by assigning textures and opacities to the well log values. The textures are spread out horizontally from the well, along the crossing horizons, for a user defined distance. For well logs, the 2D parameterization is used both for the texture parameterization and for ensuring that the textures move along the horizons outwards from the well log. See Figure 10 for examples.

The well log can be used in a depth mode to define textures that vary as a function of the well depth. This is achieved by using a synthetic well log with values that increase linearly with the depth. In this mode, the opacity of the texture transfer function defines where to texture along the depth, and the texture assignment defines which textures to use along the depth. We refer to this as a depth transfer function. In the depth mode, the user can also move the well horizontally to perform the depth varying texturing at a location that intersects the stratum that is to be texturized. The use case in Section 5.2 will give examples of depth transfer functions.

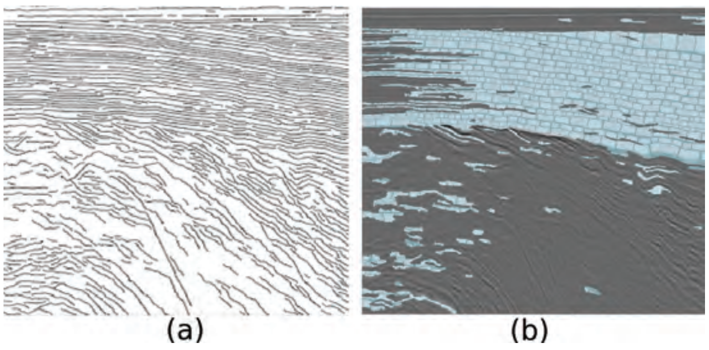


Fig. 9. a) All extracted horizon lines. b) Horizons with angles between 2 and 10 degrees are textured with a brick texture. The original seismic reflection data is shown in Figure 3.

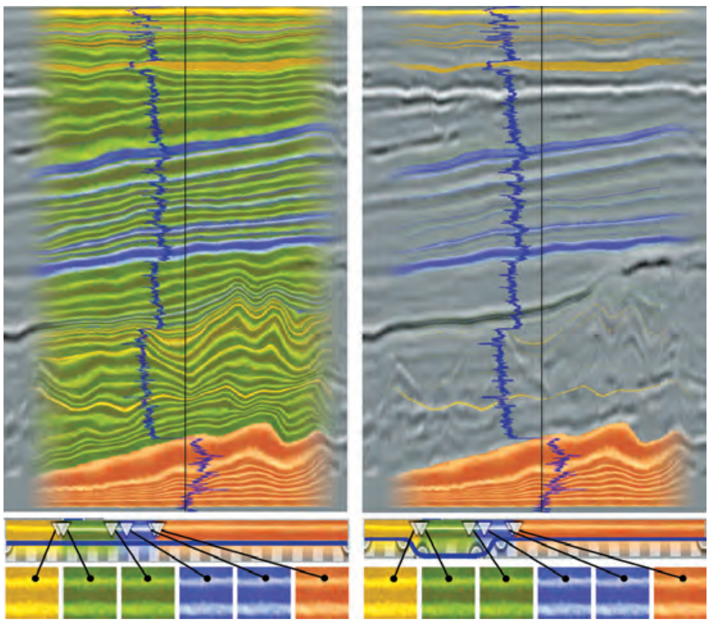


Fig. 10. Two well log transfer functions. Left: a well log with full opacity. Right: a well log with full opacity only for low and high well log values. The vertical line shows the well log path. The blue vertical graph shows the values of the well log for the well's gamma-ray radioactivity.

For derived attribute slices, the texture transfer function maps the scalar values to textures and opacities. Transparency, except for in the transitions into and out of intervals of interest, will create halos around the area of interest (Figure 11). This has the added effect that the halo thickness suggests the gradient magnitude of the attribute where thicker halos indicate smaller gradients. In general, texture transfer functions allow the interior, the transition, and the exterior areas of the values of interest to be shown in different ways, with the possibility to see underlying data.

4.2 The Line Transfer Function

An illustrative layer can also be created by using a line transfer function. The line transfer function defines lines that are curved according to the 2D parameterization of the reflection data. The colours and opacities of these lines can be linked to any derived attribute. This enables lines to describe a derived attribute by disappearing, reappearing, and changing colours. Three further line appearance parameters can be controlled globally for an illustrative layer. They are the density of the lines, the thickness of each line, and the lines' stipple repeat rate. The line density defines the minimum distance between two lines in image space. In the bottom image number 7 of Figure 12, two line layers are shown, one with blue stippled lines and one with a line partially coloured in red, yellow and black. The blue lines show the angular trend of the reflection data and they are based solely on the 2D parameterization with no relations to a seismic modality. We refer to such lines as streamlines. The opacity for the other line layer is set to transparent for low reflection intensity values. This results in one single line going through an area of high reflection intensity. The varying colours of the line arise from its line transfer function. The different zoom levels in Figure 12 show line layers with varying density and line stipple settings.

In the same way as the horizons can be assigned textures using a texture transfer function, we can assign colours and opacities to the horizon lines using a line transfer function. The user decides which horizon measure will be used by the line transfer function. Visualizing horizon lines is a method commonly used by seismic illustrators to represent the trends in the reflection data in a sparse and illustrative manner. With our approach, an illustrator can use filtering and horizon selection to draw the lines, such as seen in Figure 9a, as opposed to tracing them out manually.

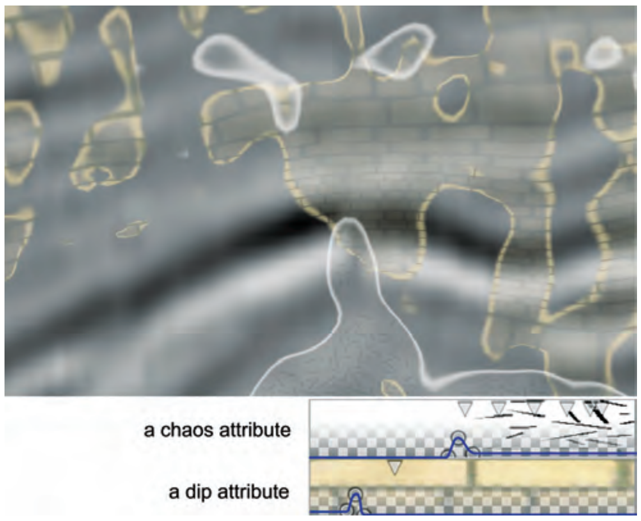


Fig. 11. The transfer functions for the derived chaos and dip attributes are defined at the bottom. The chaos attribute is transparent for low values and semi-transparent for high values with an opaque peak in between. The peak creates an opaque halo which separates low and high chaos values. A similar effect is seen for the dip attribute. The original reflection data is seen in the background.

4.3 Combining Illustrative Layers

The illustrative layers are combined into a resulting illustrative image by compositing them back to front using the over operator as described by Porter and Duff [17]. Any of the illustrative layers can be turned off to reveal the underlying layers. The user can choose the order of the layers and put the layer of highest importance in front so it is not visually obstructed by any other layer. In standard seismic illustrations, as opposed to our images, the reflectance data is not visible. We propose to integrate illustrative rendering with interpretation, therefore we enable showing the reflectance data, or other derived attributes, in the back most layer for comparison and verification reasons.

5 Results

Preprocessing the data for finding the horizons, using unoptimized Matlab code, takes from 10 to 20 minutes for a slice of size 500 by 500 samples. The parameterization takes less than a minute to calculate. The toolbox is implemented in Volumeshop [3]. Rendering requires little processing and is fast even on low end graphics cards.

In section 5.1 we present an outline of how our methods can facilitate a top-down approach for interpretation. Section 5.2 describes a use case highlighting the sketching capabilities of our tool.

5.1 Use Case: Top-Down Interpretation

Typically interpretation is performed in a time consuming bottom-up fashion with focus on details by looking at the reflection data on a fine scale. With our methods it is possible to first perform a coarse top-down interpretation. In the case a seismic survey lacks potential, a top-down approach allows for termination of the search at an early stage. This can happen as soon as a sufficient level of understanding is gained to draw conclusions. In the case of a promising survey, the top-down approach is also advantageous. At any stage, the interpretation at the current level can be used for communication purposes.

The data used in this example is a seismic reflection slice and the derived attributes chaos, dip and reflection intensity. The example is presented in Figure 12 and consists of three zoom levels. The first step in our approach is to visualize the data highly zoomed out. This gives an overview where one can identify interesting areas to investigate closer. Five zoomed out thumbnails can be seen in Figure 12, numbered one to five. The first image shows the gray reflection slice highly reduced where practically no information is left. This indicates that zoomed out overviews in the typical non-illustrative approach are not particularly useful. To get an impression of the seismic structures,

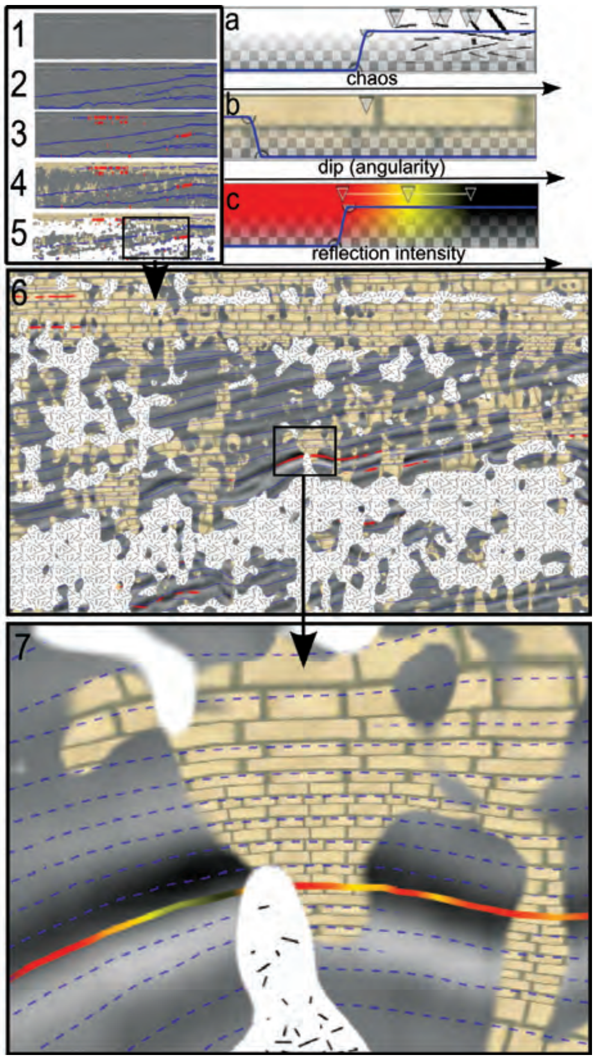


Fig. 12. A use case of an iterative drill down into the seismic reflection data. To get an overview, visual parameters are edited while looking at thumbnail-sized slices (1-5). This is followed by zooming into the data twice (5-6 and 6-7). a) is a texture transfer function on the derived chaos attribute, b) is a texture transfer function on the derived dip attribute, and c) is a line transfer function on the derived reflection intensity attribute.

we add an illustrative layer with sparse streamlines. In thumbnail image 2, the blue lines hint that the upper part of the seismic data is rather horizontal, the middle part is horizontal at the right side and angled at the left, and that the lower part is rather indecisive. For thumbnail image 3, we add another line layer with red lines in areas of high reflection intensity. Two areas with high reflection show up. We then add a brown brick texture layer in thumbnail 4 in areas with near horizontal dip. The top part of the seismic data stands out as brown. This area can be identified as a stratum with parallel seismic texture (see Figure 2). For thumbnail 5 we add another texture layer for showing chaotic areas. Two areas show up, one at the bottom and one at the left just below the parallel stratum. We now have an overview of the trends in the seismic reflection data and are ready to get a more detailed view. We zoom in on an area with low chaos and strong reflection values in image 6. At this detail level we manually adjust the texture repeat rate and the line density to get the appropriate detail level in the visualization. In image 6 one can clearly see the areas of chaos, the top stratum, and strong reflection values. Finally, in image 7, we decide to zoom in on an area with a flat spot. Flat spots are defined as areas of high reflection intensity and might indicate hydrocarbons. At this level we increase the blue line-density and render the lines in a sparse stippled style. We also increase the repeat rate for the textures further. To look closer at the variation of the reflection intensity, we add two more colours to the line transfer function. At this level one can see the different degrees of chaos by investigating the line density in the chaos texture. The interpreter can adjust the visualization, using the transfer functions and the layer ordering, so that it most closely matches his internal understanding of the data. By saving the visualization, the interpreter is able to externalize his gained internal understanding into an illustrative image that can be used as documentation. This example took about 10 minutes to interactively drill through. In this section we described how our methods support performing top-down interpretation as opposed to the existing bottom-up interpretation.

5.2 Use Case: Annotating Seismic Strata

We present a use case for interpreting stratigraphic layering in the search for hydrocarbons. This use case is created in cooperation with StatoilHydro who released the seismic data and the derived attributes to test our system. The survey had already been interpreted by the oil company but no interpretation information was given to us. Also the seismic interpreter, and the seismic illustrator, both from StatoilHydro, involved in this use case had little or no knowledge of this interpretation. The reflection data for this study is given in Figure 3. To get an initial high level overview of the data, the different strata are identified. The lithologies of the strata are unknown. The interpreter suggests the stratification as shown in Figure 13a. In the middle of the image a strong reflector is visually identified. It is thought to be a horizon separating two strata. While following the horizon from left to right the horizon splits up (yellow circle) and gives rise to two possible continuations seen as stippled lines in Figure 13a. The interpreter is uncertain whether the area between the stippled lines belongs to stratum 3 or to stratum 4. With our tool, the two stratification alternatives are sketched for the purpose of discussing them. Also, as more knowledge is gained, the textures defining the lithologies are changed.

At first, an illustrative layer is created to represent the bottom stratum 5. The stratum has a chaotic texture and it contains weak reflections due to its depth and due to strong reflectors above it. The interpreter also notices that the stratum contains reflection artifacts and concludes that the information there is not reliable. Using the parameterization of the reflection data in this area would be inappropriate since the horizons are not reliable there. Therefore, a uniform parameterization and a texture with chaotic lines is used to represent the stratum. To capture the stratum region, a depth transfer function is applied along a vertical line through the center of the image. The transfer function is set to transparent except in the depth interval where the vertical line intersects the stratum. The resulting region matches well with the interpreter's separation line seen in Figure 13b

A new illustrative layer is created and textures following the reflection parameterization are assigned to strata 1 to 4 by again using a depth transfer function. The result seen in Figure 13c matches well with the manually drawn strata lines in Figure 13a. However it was not possible to make the depth transfer function separate out stratum 2. The horizons in stratum 2 are well defined. Therefore stratum 2 is annotated by selecting its horizons by mouse picking and assigning a texture to them using a horizon transfer function. The result is seen in Figure 13d and identifies stratum 2 well. Now, one of the two alternatives of the sketch in Figure 13a is reproduced. The other alternative is quickly derived from the first alternative by moving the end-depth of the yellow texture and start-depth of the blue texture on the transfer function slightly lower (Figure 13e). The alternatives in Figure 13d and e can now be discussed among the experts. It is noticed that the fourth stratum has a somewhat distinct seismic texture (see section 1.1 for a discussion of seismic texture). It is decided to derive the discontinuity attribute with external software to see if it highlights stratum 4. If it does, then the membership of the undecided area can be resolved by comparing the discontinuity attribute to that of stratum 4. The attribute is depicted in Figure 3. By looking at the attribute one can see a distinct region in the middle. To compare this region with the annotated strata, we create a new illustrative layer with an attribute transfer function on the discontinuity attribute. Texturing only values of high discontinuity and overlaying the texturing on the illustrative layers in Figure 13e yields Figure 13f. The new illustrative layer overlaps closely with stratum 4 except for some random patches. It does not cover the undecided region. It is concluded that Figure 13e is correct since the region in question is now assumed to belong to stratum 3 but not to stratum 4. Based on the current stratigraphic mapping and due to the the strong reflection property of the horizon between stratum 3 and 4, a hypothesis is formed that stratum 4 consists of limestone. Therefore, in Figure 13g, the texturing of the fourth stratum is changed to a geological texture denoting limestone. Scrutinizing the reflection data of the stratum reveals mound structures (similar to the stratigraphic pattern 'Local Chaotic' in Figure 2). It is further hypothesized that the mounds might indicate karst bodies. Karst bodies are hollow structures created by reactions between carbonate rock and water that may act as hydrocarbon traps. An attribute is derived which is sensitive to mound like structures. To show these structures, a new illustrative layer is made. Attribute regions of high mound characteristics are displayed with one texture, and areas of medium mound characteristics are displayed with another texture (see Figure 13g and h). Figure 13h shows a zoom-in on some karst bodies. Figure 13i shows the layers with semi-transparency and opaquely emphasized strata borders and karst bodies. The interpreter now decides that this is as far as he can go with the current knowledge of the survey. This process took about half an hour and has shown that the investigated area has potential and is worth a further exploration with a detailed bottom-up interpretation. It has also saved the time of unnecessary detailed bottom-up interpretation of the topmost stippled horizon. Our tool enabled discussing possible interpretations and to arrive to conclusions at an early stage. We have also shown that the time spent creating illustrations with our system is in the order of minutes. Manually drawing illustrations of comparable quality would be in the order of hours.

There are seismic areas such as faults or noisy regions where correct automatic horizon extraction is not possible. Since the parameterization depends on the extracted horizons, the texturing will fail in these areas. As seen, even human interpreters have problems finding correct horizons in difficult areas. To address this, our system can mark out areas where parameterization fails, by using uniform texturing as discussed in Section 4.1 on an attribute that is sensitive to the problematic areas. This approach was presented in Section 5.1 for chaos areas. Ideally, textures should be discontinuous across faults. This would require the unsolved task of automatic and accurate detection of the fault surfaces. However with our methods one can use a fault sensitive attribute (there exists robust ones) and an appropriate texture pattern to mark possible fault areas where normal texturing would fail.

6 Conclusions

We have presented a toolbox with novel interpretation and rendering algorithms. It supports fast seismic interpretation and fast creation of seismic illustrations. The toolbox offers illustrative visualization, scale invariant visualization, and multi-attribute visualization. Uninterpreted seismic data has high uncertainties and fits into our quick and coarse top-down approach. Afterward, the more accurate bottom-up method is applied when higher certainty in the data has been gained. We believe our toolbox will increase the efficiency of seismic illustrators by automating time consuming tasks such as texture creation and horizon drawing. These tasks are currently performed with general drawing programs. We have informally evaluated the usefulness of our approach in 2D. We plan to extend it so that the slice plane can be positioned arbitrarily in 3D at interactive frame rates, and to integrate this approach with standard 3D volume rendering.

References

[1] Petrel seismic interpretation software, schlumberger information solutions (sis).

[2] *Federal Geographic Data Committee, Digital Cartographic Standard for Geological Map Symbolization*. Document FGDC-STD-013-2006, 2006.

[3] S. Bruckner, I. Viola, and M. E. Gröller. Volumeshop: Interactive direct volume illustration. acm siggraph 2005 dvd proceedings (technical sketch). In *ACM Siggraph Technical Sketch)*, 2005.

[4] R. Bürger and H. Hauser. Visualization of multi-variate scientific data. In *EuroGraphics 2007 State of the Art Reports*, pages 117–134, 2007.

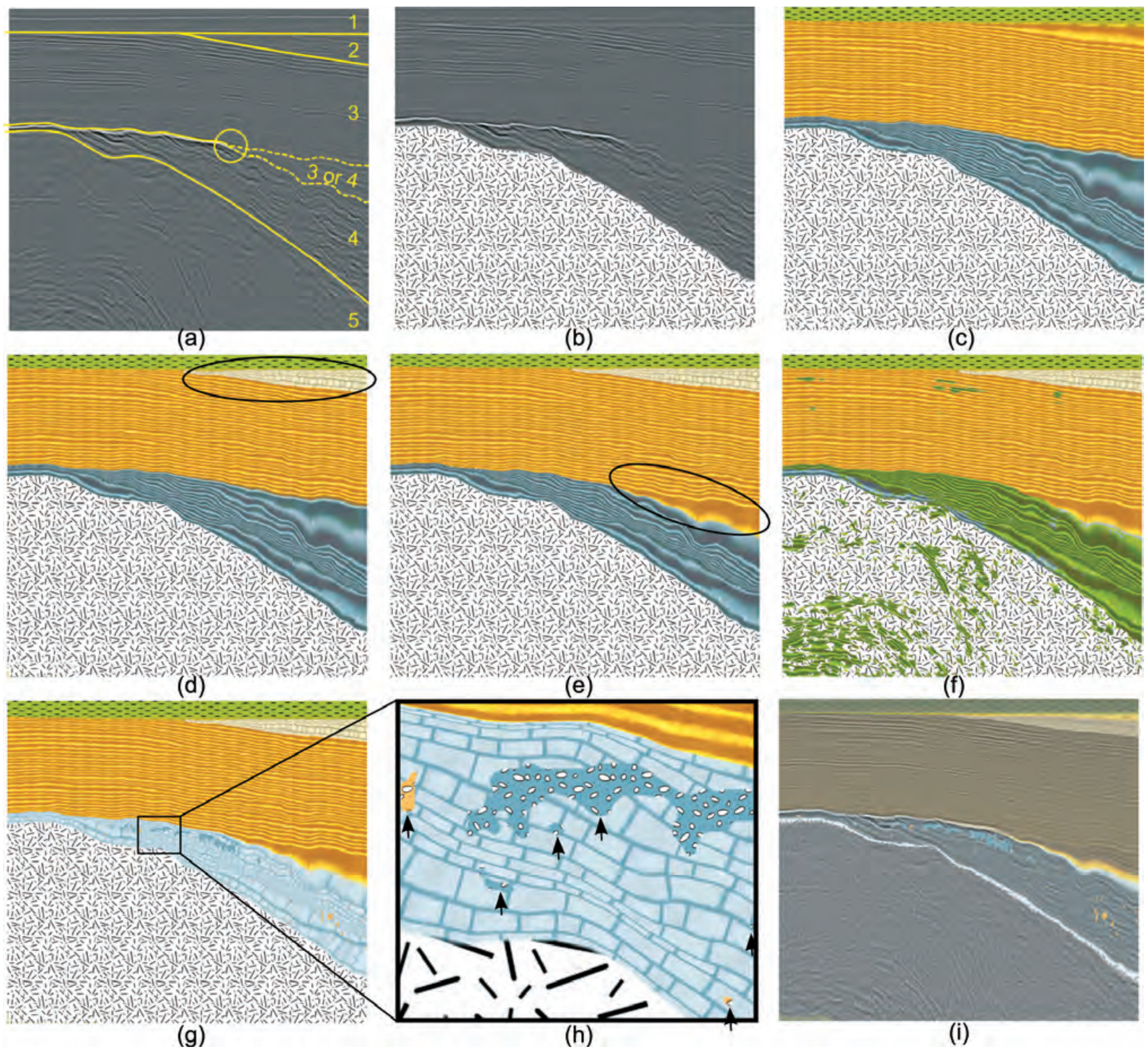


Fig. 13. Images created during an interpretation. Figure a) shows manually drawn yellow lines for separating strata. Figures b)-i) are rendered with our system. They are discussed in Section 5.2. Ellipses in (d) and (e) pinpoint the difference from the previous image. Arrows in (h) point out areas of medium (orange) and high (blue) mound characteristics.

[5] L. Castanie, B. Levy, and F. Bosquet. Volumeexplorer: Roaming large volumes to couple visualization and data processing for oil and gas exploration. *Proc. of IEEE Visualization '05*, pages 247–254, 2005.
[6] R. A. Crawfis and M. J. Allison. A scientific visualization synthesizer. In *VIS '91: Proceedings of the 2nd conference on Visualization '91*, pages 262–267, Los Alamitos, CA, USA, 1991. IEEE Computer Society Press.
[7] D. Gibson, M. Spann, J. Turner, and T. Wright. Fault surface detection in 3-d seismic data. *Geoscience and Remote Sensing*, 43(9):2094–2102, 2005.
[8] J. Grotzinger, T. H. Jordan, F. Press, and R. Siever. *Understanding Earth*. W. H. Freeman and Company, 1994.
[9] J. Hays and I. Essa. Image and video based painterly animation. In *NPAR '04: Proc. of the 3rd intn. symposium on Non-photorealistic animation and rendering*, pages 113–120, NY, USA, 2004. ACM.
[10] A. Iske and T. Randen, editors. *Atlas of 3D Seismic Attributes, Mathematics in Industry, Mathematical Methods and Modelling in Hydrocarbon Exploration and Production*. Springer, Berlin Heidelberg, 2006.
[11] W.-K. Jeong, R. Whitaker, and M. Dobin. Interactive 3d seismic fault detection on the graphics hardware. *Volume Graphics*, pages 111–118, 2006.
[12] R. M. Kirby, H. Marmanis, and D. H. Laidlaw. Visualizing multivalued data from 2D incompressible flows using concepts from painting. In *IEEE Visualization '99*, pages 333–340, San Francisco, 1999.
[13] E. M. Lidal, T. Langeland, C. Giertsen, J. Grimsgaard, and R. Helland. A decade of increased oil recovery in virtual reality. *IEEE Computer Graphics and Applications*, 27(6):94–97, 2007.
[14] D. Patel, C. Giertsen, J. Thurmond, and M. E. Gröller. Illustrative rendering of seismic data. In H. S. Hendrik. Lensch, Bodo Rosenhahn, editor, *Proceedings of Vision Modeling and Visualization*, pages 13–22, 2007.
[15] R. Pepper and G. Bejarano. Advances in seismic fault interpretation automation. In *Search and Discovery Article 40170, Poster presentation at AAPG Annual Convention*, pages 19–22, 2005.
[16] J. Plate, M. Tirtasana, R. Carmona, and B. Fröhlich. Octreemizer: a hierarchical approach for interactive roaming through very large volumes. *Proc. of VISSYM '02*, pages 53–64, 2002.
[17] T. Porter and T. Duff. Compositing digital images. *Computer Graphics Volume 18, Number 3*, pages 253–259, 1984.
[18] T. Ropinski, F. Steinicke, and K. H. Hinrichs. Visual exploration of seismic volume datasets. *Journal Proc. of WSCG '06*, 14:73–80, 2006.
[19] F. Taponecco, T. Urness, and V. Interrante. Directional enhancement in texture-based vector field visualization. In *GRAPHITE '06*, pages 197–204, New York, NY, USA, 2006. ACM.
[20] R. M. Taylor. Visualizing multiple fields on the same surface. *IEEE Computer Graphics and Applications*, 22(3):6–10, 2002.

Chapter 7

Paper III: Moment Curves

Moment Curves

Daniel Patel[*]
Christian Michelsen Research, Bergen, Norway

Martin Haidacher[†]
Institute of Computer Graphics and Algorithms, Vienna University of Technology, Austria

Jean-Paul Balabanian [‡]
Department of Informatics, University of Bergen, Norway

Eduard M. Gröller [§]
Institute of Computer Graphics and Algorithms, Vienna University of Technology, Austria

Abstract

We define a transfer function based on the first and second statistical moments. We consider the evolution of the mean and variance with respect to a growing neighborhood around a voxel. This evolution defines a curve in 3D for which we identify important trends and project it back to 2D. The resulting 2D projection can be brushed for easy and robust classification of materials and material borders. The transfer function is applied to both CT and MR data.

Index Terms: I.4.10 [Image Processing]: Image Representation—Volumetric, I.4.10 [Image Processing]: Image Representation—Statistical, I.4.6 [Image Processing]: Segmentation—Pixel classification, I.4.7 [Image Processing]: Feature Measurement—Feature representation

1 Introduction

We present a novel transfer function specification that uses the first and second statistical moments - the mean and variance. With these basic measures we are able to classify materials based on nontrivial properties, such as a material's internal variance. The transfer function is also robust to uniformly distributed noise and both identifies material boundaries and differentiates between them. The robustness of the method comes from the fact that it simultaneously considers the data on multiple scales when calculating the first and second statistical moments.

2 Related Work

There have been many publications on transfer functions and on simplifying their design. Early publications focus on 1D transfer functions where color and opacity are assigned directly to the intensity values of the volume. The design galleries by Marks et al. [8] attempt to simplify the design of transfer functions by presenting predefined transfer functions for the user to choose from. However 1D transfer functions have limited classification power. Levoy [6] introduced 2D transfer functions by additionally considering gradient magnitude. This idea was taken further by the extensive work of Kniss et al. [4]. In these works 2D transfer functions based on scalar and gradient magnitude are used to identify material boundaries which are represented as arcs in the transfer function space. Sereda et al. [11] design a transfer function for easier identification of material boundaries using the LH histogram. The LH histograms represent boundaries between materials as separable clusters of points. The separability of both the arcs in 2D transfer functions and the clusters of LH histograms require materials with low internal variance. Our approach does not. In fact high and differing variances within materials can be used to discern them. This is due to our statistical approach where we consider the properties of variably sized groups of voxels instead of single voxels in isolation. Another difference is that we identify materials based on internal material properties instead of material-material interfaces as the 2D transfer function and the LH histograms do.

Other statistical approaches use ideas related to ours. The early work by Laidlaw et al. [5] considers small neighborhoods around each voxel for calculating statistical properties used for classification. Sato et al. [10] calculate a set of derived features for identifying local structures. Caban and Rheingans [1] divide the data into blocks and calculate a wide range of textural properties for each block. Similarly in the work of Lundström et al. [7], histograms of blocks are considered to improve material classification in medical data. All of these works use static neighborhoods and often calculate complex neighborhood properties. We use a dynamically changing neighborhood to better find an optimal set of voxels for calculating properties on. This approach gives good results with noncomplex neighborhood properties.

Both the works by Hadwiger et al. [3] and Correa and Ma [2] deal with size transfer functions that classify regions based on their sizes. Both methods require a preclassification of the data for defining the regions to apply the size transfer functions. For preclassification Hadwiger et al. perform region growing and calculate the volumetric size of regions whereas Correa and Ma perform diffusion by Gaussian smoothing to calculate region thicknesses. Correa and Ma also include a good overview of earlier work in scale-space analysis.

3 The theory of Moment Curves

For each voxel in a volume we calculate a sequence of values based on the voxels in its vicinity. We then assign optical properties to the voxel based on this sequence of values. This results in a classification of materials or material boundaries. To calculate the sequence of values for a voxel v, we consider all its neighbor voxels that are inside a sphere centered at v with radius r (see Figure 1). For each sphere of radius r, we calculate the mean and the variance of the voxels inside. With r increasing from zero to a predefined maximum value we get a sequence of (mean μ, standard deviation σ) pairs for the voxel v. We depict this sequence of the first and second moment parameterized by r, as a curve in 3D space. As seen to the right in Figure 1 the axes of the 3D space are the mean value μ in red, the standard deviation σ in green, and the radius r in blue. Each voxel in the volume is thus represented as a curve in 3D. We call the curves moment curves since they describe the evolution of the first and second statistical moments as the neighborhood increases. By selecting certain groups of curves we end up with a classification of voxels. In the following paragraphs we describe how the shape of these curves relate to material properties.

We demonstrate the properties of the moment curves by considering a simple dataset consisting of two materials planarly touching (see Figure 2) and having infinite extent on their respective sides. The intensity values of each material are normal distributed with defined means and variances. The sequence of mean and standard deviation pairs for voxels inside a material will estimate the material's mean and standard deviation with increasing accuracy as the radius grows. We will now separately consider how the standard deviation and how the mean of a voxel evolves as the radius increases. Figure 2 shows the two materials and voxels at different positions in

[*]e-mail:daniel@cmr.no
[†]e-mail:haidacher@cg.tuwien.ac.at
[‡]e-mail:Jean-paul.Balabanian@ii.uib.no
[§]e-mail:groeller@cg.tuwien.ac.at

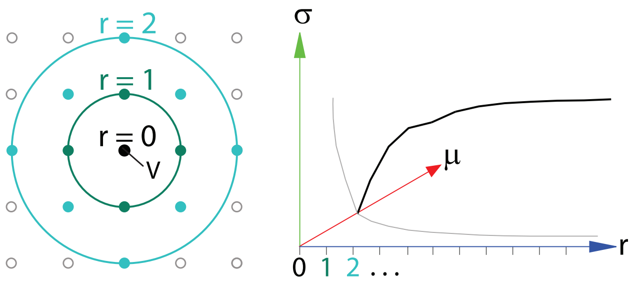


Figure 1: Left: The neighbor voxels for different radii of a voxel v. Right: A moment curve in 3D space as a function of radius r. The gray lines are the projections of the curve into the $r = 0$ plane and the $\sigma = 0$ plane.

the materials. For some voxels several circles are shown to indicate varying radii.

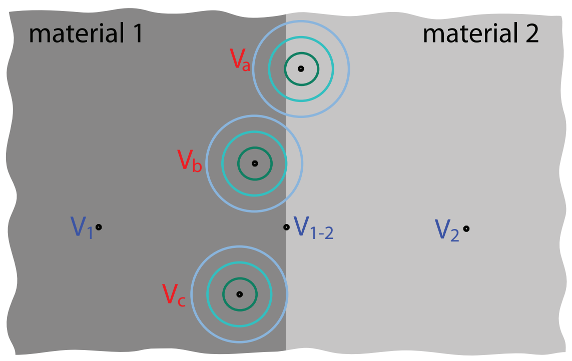


Figure 2: Two materials with different distributions touching each other.

For voxel V_1 we have plotted the evolution of the standard deviation for increasing radii. This is the curve labeled V_1 in Figure 3. As soon as the neighborhood is large enough, the curve stabilizes at the standard deviation of the material. We have colored the part of the curve after stabilization in blue and the part before in gray. A single sample has no standard deviation so all curves start at $\sigma = 0$ for $r = 0$. As the neighborhood increases, the standard deviation increases until it stabilizes. At what radius the curve stabilizes depends on the 'patchiness' of the material. The patchier it is, the higher radius it stabilizes at. A corresponding curve is plotted for voxel V_2 which is in material 2. Material 2 has lower variance than material V_1 therefore the curve stabilizes below the curve for V_1.

In Figure 4 the curve for voxel V_1 is drawn with its mean value as a function of the radius. Also here, for large enough radii, the curve stabilizes at the mean value for the material. However when the radius is 0, the mean will be the intensity of the voxel itself which can be any value from the material's normal distribution. The range of the values of material 1 is drawn as a thick vertical gray line along the μ axis. The material's distribution is drawn as well. The curve for voxel V_1 may thus start at any point along this range with a probability corresponding to the material's distribution. From the starting point, the curve will move towards the mean of the material. This is represented by a gray fan-in from the material range into the blue stabilized curve. The μ curve for voxel V_2 in material 2 is also drawn. Material 2 has a higher mean value than material 1. The distributions of the materials are overlapping. This is depicted in the overlapping vertical lines along the μ axis. It is thus possible that a curve for a voxel in material 1 intersects a curve for a voxel

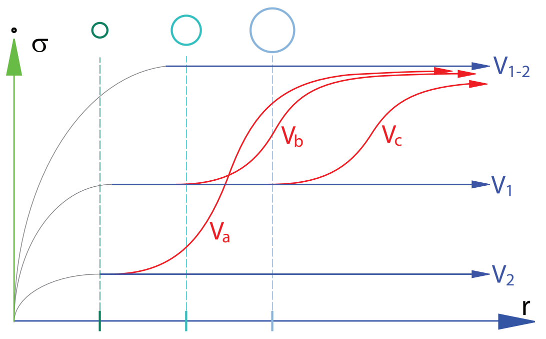


Figure 3: The standard deviation of the voxels in Figure 2 as a function of increasing radius.

in material 2 before the curves stabilize. This is represented by intersecting lines from the fan-ins of the two curves.

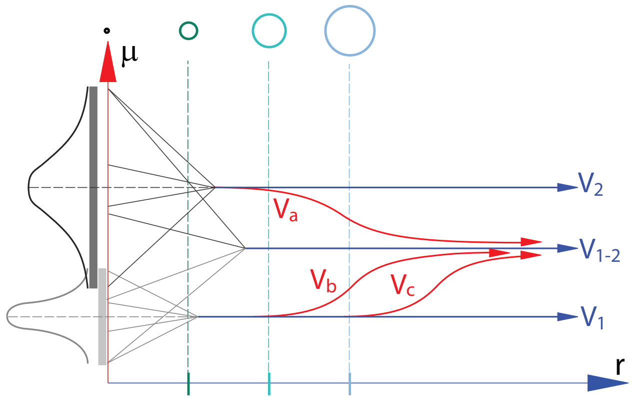


Figure 4: The mean values of the voxels in Figure 2 as a function of increasing radius.

Considering the stabilized parts of the μ and σ curves, one can separate materials with differing means or standard deviations. In contrast 1D transfer functions cannot discern materials solely based on variance and cannot binary discriminate materials with overlapping intensity distributions. Note that our curves include the 1D transfer function as a special case when assigning optical properties to voxels based on their mean values at zero radius.

Now we will consider the μ and σ curves of a voxel V_{1-2} situated exactly at the border between the two materials. As the radius increases, an equal number of samples from each of the two material distributions is included. Therefore the mean will stabilize exactly between the mean for material 1 and material 2 as shown in Figure 4. The distribution for the neighborhood of V_{1-2} will be bimodal since it consists of the sum of the distributions from each material. Therefore the fan of the V_{1-2} curve covers both the material ranges. The standard deviation for curve V_{1-2} is seen in Figure 3. This curve will stabilize at a standard deviation which is less intuitive to understand than the mean. We will give an analytical expression for this further on.

In Figure 2 we have drawn voxels V_a, V_b and V_c having different distances to the border. Their corresponding σ and μ curves are drawn in Figure 3 and Figure 4 respectively. Three circles with different radii are drawn around each of the voxels and these circles are also marked along the radius axes in the two plots. From a specific radius onward, the neighborhood will include voxels from the neighboring material. The μ and σ curves will then move away from the stable part and will asymptotically approach the respective mean and standard deviation of V_{1-2}. This happens because the number of samples from material 1 and material 2 will equalize as the radius goes towards infinity. We will now present expressions of the mean and of the variance as a function of the ratio κ of samples between material 1 and 2. Let the two materials have mean and standard deviation (μ_1, σ_1) and (μ_2, σ_2) respectively. The derivation of these equations are given in the Appendix.

$$\mu_\kappa(\kappa) = (1-\kappa)\mu_1 + \kappa\mu_2 \tag{1}$$

$$\sigma_\kappa^2(\kappa) = -(\mu_1-\mu_2)^2\kappa^2 + \left((\mu_1-\mu_2)^2 - ({\sigma_1}^2 - {\sigma_2}^2)\right)\kappa + {\sigma_1}^2 \tag{2}$$

The mean is a linear interpolation between the μ's weighted by κ, and the variance is a second degree polynomial in κ where the coefficents are expressions of the σ's and μ's.

We will now present the relation between a border voxel at different distances to the border of the two materials and the μ and σ for the samples inside a fixed radius around that voxel. The materials can then be represented as two points in a diagram with μ on the horizontal and σ on the vertical axis. As the border voxel moves from one side of the border to the other, the ratio of voxels inside a neighborhood of fixed radius will go from material 1 ($\kappa = 0$) to material 2 ($\kappa = 1$). By using the ratio as an argument into equation 1 and equation 2, the $(\mu_\kappa, \sigma_\kappa^2)$ point for the border-voxel will move on a parabolic arc from material 1 to material 2. The square root of a second order polynomial with negative factor of the second order term, as given in equation 2, is an ellipsoid. Therefore, the $(\mu_\kappa, \sigma_\kappa)$ point will move from material 1 to material 2 along an ellipsoid as the ratio changes. Exactly at the border, the ratio κ equals 0.5 and the mean and variance is here:

$$\begin{aligned} \mu_{1-2} &= \mu_\kappa\left(\tfrac{1}{2}\right) = \tfrac{\mu_1+\mu_2}{2} \\ \sigma_{1-2}^2 &= \sigma_\kappa^2\left(\tfrac{1}{2}\right) = \left(\tfrac{\mu_1-\mu_2}{2}\right)^2 + \tfrac{{\sigma_1}^2+{\sigma_2}^2}{2} \end{aligned} \tag{3}$$

When knowing the mean and variance of two bordering materials, equations (3) allow to calculate the shape of the arc that appears for voxels near the material's border. One can then automatically create a transfer function that performs a brushing which classifies this border. We showed earlier that voxels inside materials have moment curves that stabilize. We have now shown that voxels on the boundaries between two planarly touching materials also have stabilized curves. For voxels close to boundaries the moment curves switch at increasing radius from the material moment curve to the border moment curve. The radius where the transition happens indicates the distance of the voxel to the boundary. By selecting voxels that have stabilized curves which switch at a certain radius, we can identify regions with specific distances to boundaries.

4 Synthetic Data

We will now demonstrate the properties of the moment curves on a 3D test dataset consisting of 6 cylinders, each with a radius of 32 voxels, a height of 40 voxels and noise added so as to get different standard deviations. The cylinders are named cyl_σ where σ is the standard deviation in the interior.

The top left image of Figure 5 shows a slice through the cylinders. In Figure 6 we have plotted 25000 moment curves from the interior of cyl_2 and cyl_{20}. The curves are projected onto the right side and the bottom of the plot. The gray semitransparent rectangle is a cutting plane at $r = 1$. Red points are drawn on the cutting plane at positions where the moment curves intersect. One large cluster of points can be seen on this plane for cyl_{20} and one small cluster for cyl_2. The green curve shows the curve for one particular voxel in cyl_{20}. One cluster of curves for each material is generated and each cluster converges to the mean and standard deviation of the corresponding material. The curves for the materialin cyl_2 with the low standard deviation converge faster and voxels in cyl_{20} have values from the whole intensity interval.

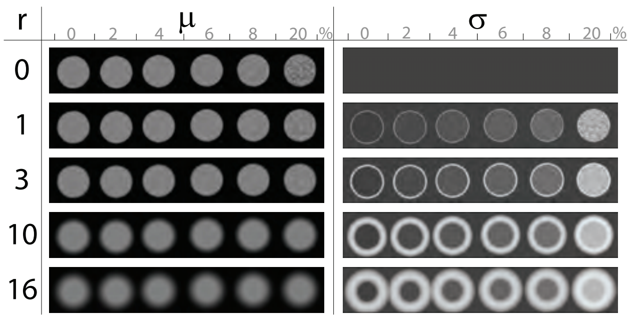


Figure 5: To the left μ and to the right σ for voxels with increasing radii are depicted. All cylinders have a $\mu = 0.5$ and an increasing $\sigma = \{0\%, 2\%, 4\%, 6\%, 8\%, 20\%\}$ from left to right.

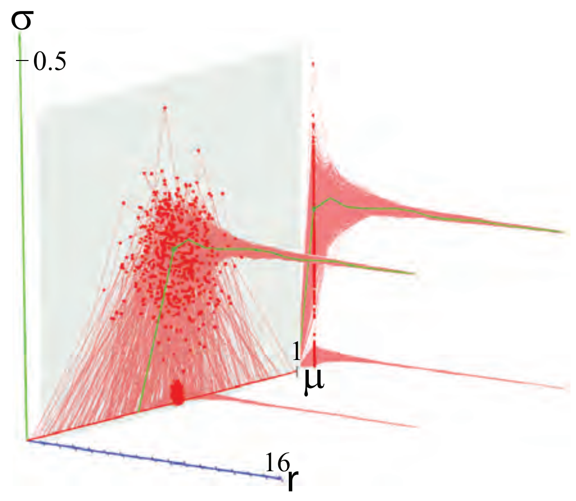


Figure 6: Moment curves, projections and $r = 1$ intersections for cyl_2 and cyl_{20} in Figure 5.

Now we will consider how moment curves of voxels close to material borders behave in practice. In Figure 7b we have drawn 31 moment curves for voxels along a line from the interior of cyl_8 to the exterior (see Figure 7c). Figure 7a and 7d show the projection of moment curves onto the r-μ and the r-σ plane. The green curve represents the voxel closest to the border. Notice how curves jump out of their stabilized path as soon as the neighborhood is large enough to cross the border. The moment curve of the border voxel is stabilizing fastest and the other curves, according to their distance to the border, are asymptotically moving towards it. In Figure 7b) we show a cutting plane at $r = 16$ with curve intersections marked as red dots. Since we know the σ's and μ's of each material, we can get an expression for μ and σ for voxels moving from material 1 to material 2. By substituting $\mu_1 = 0.5$, $\sigma_1 = 0.08$ for cyl_8 and $\mu_2 = 0$, $\sigma_2 = 0$ for the exterior into equation 2 we get $\mu_\kappa = 0.5\kappa$ and $\sigma_\kappa^2 = -0.25\,\kappa^2 + 0.2564\,\kappa$. The analytic $(\mu_\kappa, \sigma_\kappa)$ path overlaid on the intersection points at $r = 16$ is depicted in Figure 8. The tight overlap demonstrates the expected correspondance between the analytic formulas and the discrete samples of the test dataset.

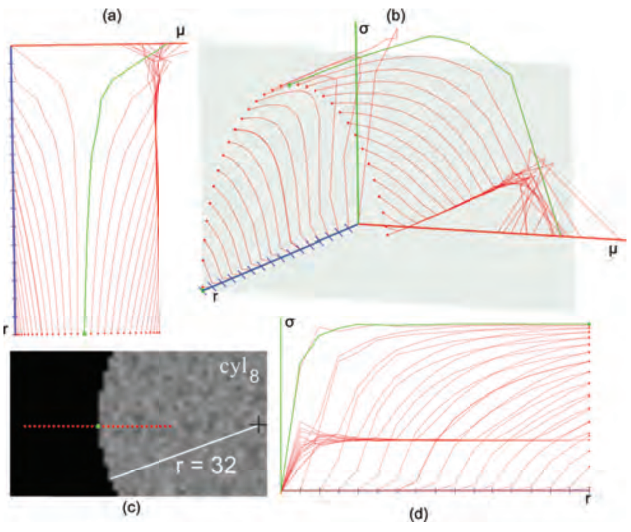


Figure 7: Moment curves for voxels along a line from the inside to the outside of cyl_8. The green curves and point represent a voxel on the boundary of the material.

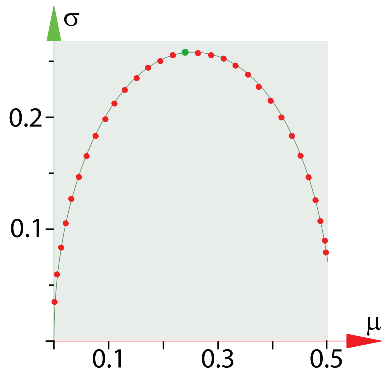


Figure 8: The curve intersections at $r = 16$ as red dots and an analytically plotted (μ_κ, σ_κ) path overlaid to show correspondence.

We have now discussed the different types of moment curves that occur in our dataset. In Figure 9a we plot moment curves representing voxels from the entire volume. One can clearly see clusters for each cylinder and arcs from each material to their surrounding background values. We apply a transfer function on the moment curves to classify the cylinders and their borders. First we try to classify cyl_8 by defining a blue rectangle on a cutting plane at $r = 16$. The rectangle is located around the cluster for cyl_8. Our transfer function is defined such that each voxel having a moment curve that intersects the rectangle at $r = 16$ will be classified and colored. The result can be seen at the bottom of Figure 9a.

The interior voxels of cyl_8 which have a distance of more than r=16 voxels to the border are now classified in blue. Voxels with distance less than 16 to the border are not classified because their curves will have 'jumped' away from the stabilized path at earlier radii. The transfer function also classifies circles inside each of the cylinders of lower variance. The over-classified voxels are due to moment curves which have 'jumped' away from their stabilized part and intersect the rectangle as they move toward the top of the arcs between material and exterior. We can avoid classifying these regions by only classifying curves that are stabilized as they intersect the rectangle. The degree of stabilization of a moment curve can be expressed by its partial derivative along r. We define the derivative curve as having it's μ' value for a certain radius r equal to the difference between the μ at $r-1$ and r. Similarly we find the curve's σ' value for a radius r to be the difference between the σ at r-1 and r. These derivative curves can be visualized in the same manner as the moment curves. In Figure 9b we are visualizing all the derivative curves. This yields an interesting pattern where the curves seem to move along a cylinder which has a decreasing thickness along the r axis.

We can select moment curves with an arbitrary degree of stabilization by brushing on the derivative curves. In Figure 9c the (μ',σ') pairs for each curve intersecting $r = 16$ can be seen. The blue rectangle defines the curves we want to brush to select stabilized moment curves.

In Figure 9d the moment curves corresponding to the points brushed in the derivative space are shown in blue. Only stabilized curves are in blue and these stabilized curves are found in the interior of each material, at the border between each material and background, and in the background itself. We now assign blue to any voxel having a stabilized curve according to the brush in the derivative space in Figure 9d. The transfer function is seen as a rectangle covering all moment curves of the data. By selecting all stabilized curves we get the interiors and the borders of the materials as well as the background (see bottom of Figure 9d). Close to the borders safety margins of not-selected voxels appear. The width of these margins depends on the chosen radius r.

Now we continue to refine the transfer function with different colors for each easily separable cluster of stabilized curves. The transfer function is defined as a set of coloured polygons in the plane at $r = 16$ (top of Figure 9e). The corresponding classification is seen at the bottom of Figure 9e.

After brushing the curves we are able to select for example only cyl_8. In the top row of Figure 10 we do a volume rendering of the test dataset using a semitransparent 1D black-to-white transfer function and with cyl_8 classified as blue. Due to the safety margin the classified cylinder is too small. To compensate for the shrinking, we dilate the volume r voxels where r is the radius of the plane the transfer function was defined on. In the bottom row of Figure 10 an image series of the dilation steps for the cylinder can be seen. At the last step it almost covers its original space again. However the edges are rounded, so a perfect reconstruction has not been made. By defining transfer functions at cutting planes with lower radii one gets less rounding of edges. This has the disadvantage of brushing less stabilized curves since the maximal radii considered are smaller.

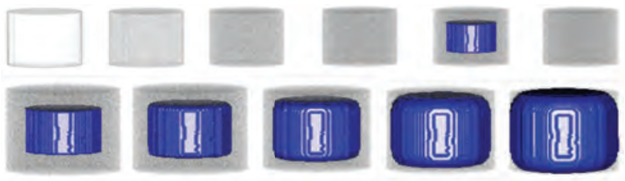


Figure 10: Top row shows volume rendering with cylinder cyl_8 classified. The bottom row shows the dilation steps for the cylinder.

We propose the following workflow for the transfer function design. First select a radius where most materials have stabilized curves ($r = 16$ in our example). Visualize the intersection of the curves with a cutting plane at the selected radius. This reduces the 3D space to a 2D plane where it is easier to define a transfer function. Depending on the data, the 2D scatterplot can be cluttered and contain many arcs. Stabilized paths of the scatterplot can

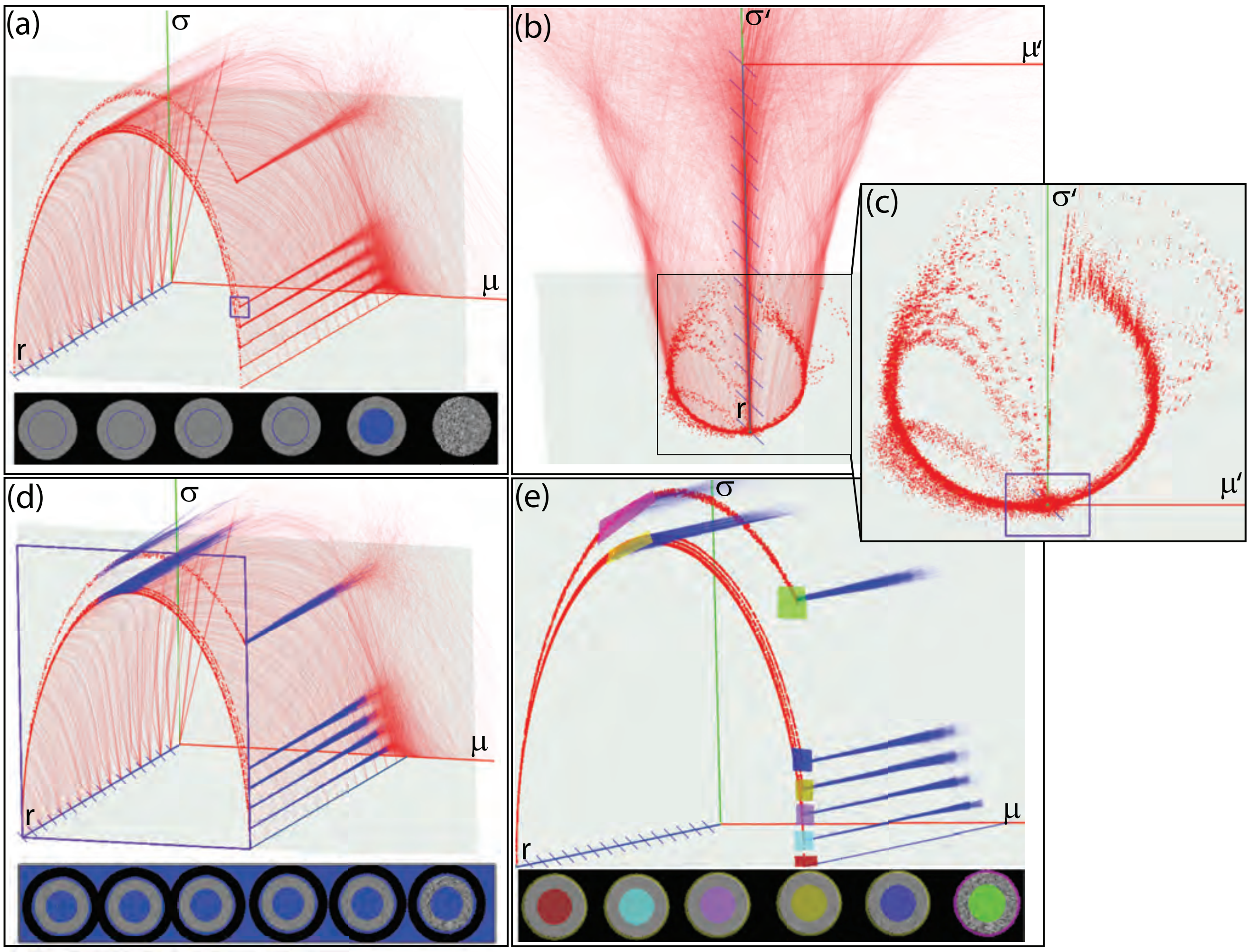


Figure 9: (a) The moment curve space of the cylinder test data with a blue brush rectangle and the corresponding classification below. (b) Derivative curves from the moment curves. (c) The derivative scatterplot of the moment curves intersecting the plane $r = 16$. (d) The curve space of the test dataset with stabilized curves shown in blue. A derivative brush as defined in c) selects all stabilized curves. The result is a classification of all material interiors, material borders, and background. (e) A color transfer function on stabilized moment curves identifies materials of different variances and their borders.

be identified by brushing the corresponding derivative scatterplot (Figure 9c). Now one can assign colors to the different clusters of stabilized curves. Figure 11a shows the projected moment curves in red and the stabilized clusters in blue. Figure 11b shows only stabilized curves yielding a clear view with easily identifiable regions. The zoom-in corresponds to the material boundaries of $cyl_{0,2,4,6,8}$.

5 Implementation

The classification procedure is performed in four steps. Firstly the user selects an area on a slice positioned through the volume. The moment curves for the voxels in that area are calculated on the CPU and visualized. Curves for areas on other slices can be added. Secondly the user performs a brushing on the moment curves. A classification is then performed based on the brush on all samples on the slice. By exploiting the parallelity of the GPU we are able to do this in real time. The frame rate is sensitive to the maximum calculated radius. The number of voxels being considered for each fragment of the slice increases cubically as a function of the radius. Using precomputed means and variances could considerably reduce the classification time but at the expense of texture memory. Thirdly, as soon as the user finds the slice classification satisfactory, a 3D classification volume is built by iteratively classifying each slice in the volume. On an Nvidia Geforce 8800 GTS graphics card this can take from a few seconds on small datasets up to a minute on the CT dataset of size $512 \times 512 \times 1112$, as used in the results Chapter. Finally, a distance transform up to a distance of r voxels on the classified volume is performed. Afterwards we render in real-time the classified regions and dilate them up to r voxels by changing the isovalue of the region borders. The distance field from the dilation step is additionally used in calculating normals for Phong shading the classified regions. Our method is different from seeding and growing approaches for data segmentation since all voxels are classified independently from each other. Also the method is global and therefore does not require a specific seeding.

6 Results

We have applied our transfer function successfully on a CT dataset of a human with a resolution of 512x512x1112 and on an MR dataset of a sheep heart with a resolution of 352x352x256. We follow the same approach as suggested in Chapter 4 for the cylin-

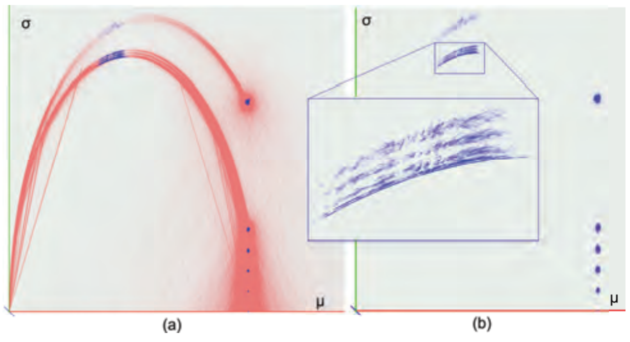


Figure 11: (a A 2D projection of the 3D moment curve space with identified stable regions in blue. (b Only stabilized curves are shown with a zoom-in on the boundaries.

der test dataset. The steps for analyzing the CT torso can be seen in Figure 12. The moment curves of 30000 evenly spaced samples taken on the leftmost slice in 12g are shown in 12a. To find stabilized curves we calculate the derivative curves 12b and brush (small blue rectangle) in the r=16 cutting plane in 12c. After brushing the stabilized moment curves are visualized in blue in 12d. For easy brushing we use the 2D projection of all curves in 12e. In 12f we define coloured brushes on easily identifiable regions. The resulting classification for three different slices is shown in 12g. Muscles are colored green, fat is colored blue, the liver is red, and the spleen is bright cyan. Also some voxels in the lungs are classified separately as dark cyan. Finally the transition between fat and muscles is pink. We dilate the spleen and liver and volume render them in Figure 13. For the background material a semitransparent 1D black-to-white transfer function is used.

The second dataset we use is an MRI of a sheep heart. This dataset is challenging to create meaningful classifications and has therefore been used in transfer function papers as a benchmark. In the paper by Pfister et al. [9] several 1D transfer function variants as well as the 2D transfer function by Kniss et al. [4] (Figure 14a) were tested on this dataset. Also the LH histogram has been applied on this dataset by Sereda et al. [11](Figure 14b). From the comparison in Pfister et al. [9] and even by quickly experimenting with a 1D transfer function it follows that a 1D transfer function only manages to separate the background from the sheep heart but fails to clearly separate interior materials in the heart. The LH histogram [11] attempts to separate a higher valued area in the middle of the heart from the rest. However by looking at the LH transfer function one can see that the yellow and red part extend over almost the entire F_H domain and therefore basically define a 1D transfer function which will fail to separate the middle part from the rest. Also the 2D transfer function by Kniss et al. [4] used in Pfister et al. [9] has a continuous color gradient and thus does not clearly separate the two regions. Our approach also failed to classify the middle region. This is partly due to the small size of the dataset with moment curves that cannot achieve a big enough radius to stabilize before they grow out of the region. It is also due to a lack of homogeneity of the middle region. The darker colored middle region has bright thin structures going through it. This disables curve stabilization. To address these issues we apply a two level approach. First we define a simple 1D transfer function that classifies the middle region but also over-classifies some areas outside it. Then we calculate the moment curves by only considering voxels with values in the interval defined by the 1D transfer function. We have defined the 1D transfer function to exclude the thin structures that were inside the region we try to classify. Then also the neighboring materials are excluded. Now the moment curves have a better chance to stabilize. With this technique we were able to classify the middle region as can be seen in Figure 14c. The slices in Figure 14d-h show the classified areas in green and the areas within the 1D transfer function interval in red. We believe we have comparable or better results with our approach than earlier published results as can be seen in the slices and the volume renderings.

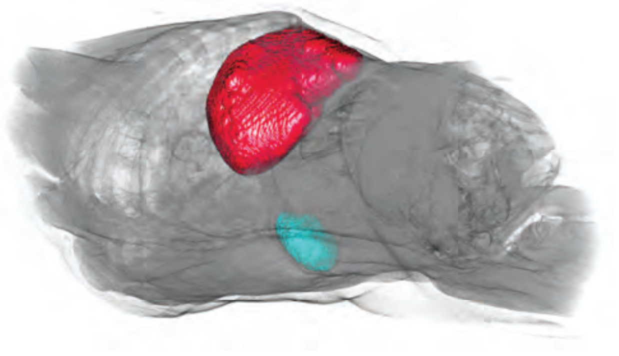


Figure 13: Two classified and dilated regions in a human CT dataset.

7 Conclusion and Future Work

We have introduced moment curves and showed how they behave in different material arrangements. We have described a workflow for applying a specific transfer function to classify materials and material boundaries. The necessary transfer function editing is performed in 2D on easily identifiable clusters. Finally we have applied our approach on a CT dataset of a human torso and classified separate tissue types. We have also successfully applied a slightly modified two-level version of the transfer function on MR data and compared it with existing transfer functions. We believe moment curves has potential on other types of data as well. By combining our method with an automatic cluster-identification algorithm, one can imagine a fully automatic transfer function specification. The stabilized moment curves for a volume would be calculated. The cluster-identification algorithm would segment each cluster. In the successive rendering users could assign optical properties or tags to each segmented object.

Acknowledgements

The authors wish to thank Dr. Ruben Patel, Professor Helwig Hauser and Professor Ivan Viola for fruitful discussions and also Dr. Christopher Giertsen for supporting this work.

References

[1] J. Caban and P. Rheingans. Texture-based transfer functions for direct volume rendering. *IEEE TVCG*, 14(6):1364–1371, 2008.

[2] C. Correa and K.-L. Ma. Size-based transfer functions: A new volume exploration technique. *IEEE TVCG*, 14(6):1380–1387, 2008.

[3] M. Hadwiger, F. Laura, C. Rezk-Salama, T. Hollt, G. Geier, and T. Pabel. Interactive volume exploration for feature detection and quantification in industrial CT data. *IEEE TVCG*, 14(6):1507–1514, 2008.

[4] J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional transfer functions for interactive volume rendering. *IEEE TVCG*, 8(3):270–285, 2002.

[5] D. H. Laidlaw, K. W. Fleischer, and A. H. Barr. Partial-volume bayesian classification of material mixtures in MR volume data using voxel histograms. Technical report, California Institute of Technology, Pasadena, CA, USA, 1997.

[6] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8(3):29–37, 1988.

[7] C. Lundström, P. Ljung, and A. Ynnerman. Local histograms for design of transfer functions in direct volume rendering. *IEEE TVCG*, 12(6):1570–1579, 2006.

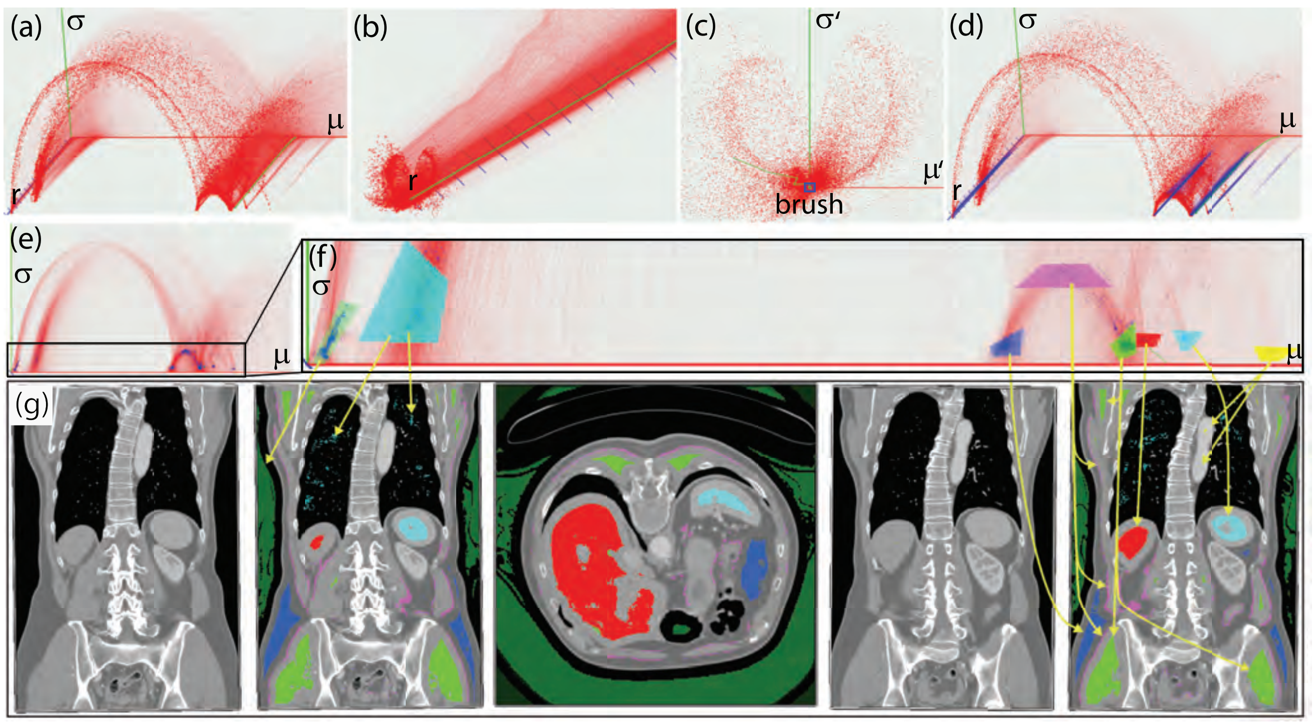


Figure 12: Moment curve steps a)-f) for classifying regions of a human CT dataset. Results show in e) two coronal slices before and after classification and one classified axial slice.

[8] J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber. Design galleries: a general approach to setting parameters for computer graphics and animation. In *SIGGRAPH '97*, pages 389–400, 1997.

[9] H. Pfister, C. Bajaj, W. Schroeder, and G. Kindlmann. The transfer function bake-off. *VIS '00: Proceedings of the 11th IEEE Visualization 2000*, pages 523–526, 2000.

[10] Y. Sato, C.-F. Westin, A. Bhalerao, S. Nakajima, N. Shiraga, S. Tamura, and R. Kikinis. Tissue classification based on 3D local intensity structures for volume rendering. *IEEE TVCG*, 6(2):160–180, 2000.

[11] P. Sereda, A. Bartroli, I. Serlie, and F. Gerritsen. Visualization of boundaries in volumetric data sets using LH histograms. *IEEE TVCG*, 12(2):208–218, 2006.

APPENDIX

In this section equations (1) and (2) of Chapter 3 are derived.

The equations for calculating the mean and variation for a discrete collection of samples $\{x_1, x_2, ..x_n\}$, are given in (4) and (5):

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4}$$

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i-\mu)^2 = \frac{1}{n}\sum_{i=1}^{n}x_i^2 - \mu^2 \tag{5}$$

Assume a material 1 has n_1 samples indexed $\dot{x}_i$ where the mean of the samples are μ_1 and the variance is σ_1^2. Similarly, assume a material 2 with n_2 samples $\ddot{x}_i$ having mean μ_2 and variance σ_2^2. We want to find the mean μ_3 and variance σ_3^2 for a material 3 consisting of the intensity values of material 1 and 2: $\{\dddot{x}_1, \dddot{x}_2, ..., \dddot{x}_{n_3}\} = \{\dot{x}_1, \dot{x}_2, ..., \dot{x}_{n_1}, \ddot{x}_1, \ddot{x}_2, ..., \ddot{x}_{n_2}\}$. We have:

$$n_3 = n_1 + n_2 \tag{6}$$

Since addition is associative we have:

$$\sum_{i=1}^{n_3}\dddot{x}_i = \sum_{i=1}^{n_1}\dot{x}_i + \sum_{i=1}^{n_2}\ddot{x}_i \tag{7}$$

$$\sum_{i=1}^{n_3}\dddot{x}_i^2 = \sum_{i=1}^{n_1}\dot{x}_i^2 + \sum_{i=1}^{n_2}\ddot{x}_i^2 \tag{8}$$

We now want to express μ_3 and σ_3 as a function of μ_1, μ_2, σ_1, σ_2, n_1 and n_2. By rearranging (4) for material 1 and 2 we get:

$$\sum_{i=1}^{n_1}\dot{x}_i = n_1\mu_1 \qquad \sum_{i=1}^{n_2}\ddot{x}_i = n_2\mu_2 \tag{9}$$

Similarly by rearranging (5) we get:

$$\sum_{i=1}^{n_1}\dot{x}_i^2 = n_1(\sigma_1^2+\mu_1^2) \qquad \sum_{i=1}^{n_2}\ddot{x}_i^2 = n_2(\sigma_2^2+\mu_2^2) \tag{10}$$

For μ_3, substituting (6) and (7) into (4), we get the expression in the middle of (11). By applying (9) on the middle expression we get the right side expression of (11):

$$\mu_3 = \frac{1}{n_3}\sum_{i=1}^{n_3}\dddot{x}_i = \frac{1}{n_1+n_2}\left(\sum_{i=1}^{n_1}\dot{x}_i + \sum_{i=1}^{n_2}\ddot{x}_i\right) = \frac{1}{n_1+n_2}(n_1\mu_1 + n_2\mu_2) \tag{11}$$

For the variance σ_3 we substitute (10) into (8) and apply the result on (5) to get the right hand of the first line in (12). Then by further substituting (6) and (11) we get the second line in (12):

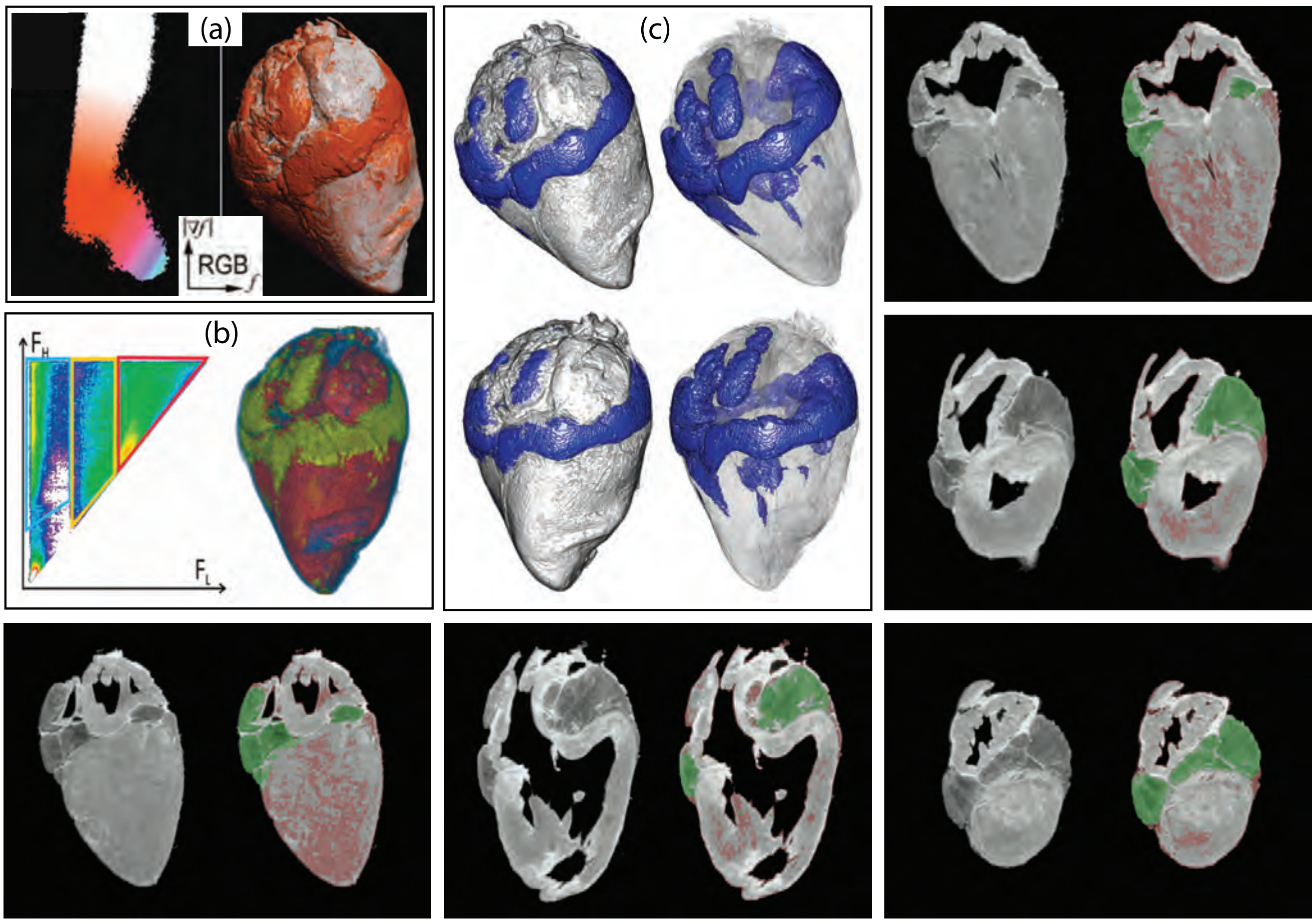


Figure 14: Comparison of (a) 2D transfer function, (b) LH histogram and (c) moment curve classification. Renderings to the right in (c) are semitransparent. Unclassified and classified moment curve slice pairs are also shown.

$$\sigma_3^2 = \tfrac{1}{n_3}\sum_{i=1}^{n_3} \dddot{x}_i^2 - \mu_3^2 = \tfrac{1}{n_3}(n_1(\sigma_1^2+\mu_1^2)+n_2(\sigma_2^2+\mu_2^2)) - \mu_3^2$$
$$= \tfrac{1}{n_1+n_2}(n_1(\sigma_1^2+\mu_1^2)+n_2(\sigma_2^2+\mu_2^2)) - (\tfrac{1}{n_1+n_2}(n_1\mu_1+n_2\mu_2))^2 \tag{12}$$

We do not want to specify both n_1 and n_2 but rather the ratio κ between them. $\kappa = 0$ means only samples from material 1, $\kappa = 0.5$ means an equal number of samples from both materials and $\kappa = 1$ means only samples from material 2 are taken. We therefore express n_1 and n_2 as a function of κ and n_3:

$$n_1 = n_3(1-\kappa), \quad n_2 = n_3\kappa \tag{13}$$

A material 3 with ratio κ between samples from material 1 and samples from material 2 will thus have the mean $\mu_3(\kappa)$ from inserting (13) into the right hand expression of (11):

$$\mu_3(\kappa) = \frac{1}{n_3}(n_3(1-\kappa)\mu_1 + n_3(1-\kappa)\mu_2) \quad = \quad \boxed{(1-\kappa)\mu_1 + \kappa\mu_2} \tag{14}$$

This concludes the derivation of equation (1) in Chapter 3.

Similarly, a material 3 with ratio κ between samples from material 1 and samples from material 2 will have the variance σ_3^2 from inserting (13) into (12) and using (14) for the mean:

$$\sigma_3^2(\kappa) = \frac{1}{n_3}\left(n_3(1-\kappa)(\sigma_1^2+\mu_1^2) + n_3\kappa(\sigma_2^2+\mu_2^2)\right) - ((1-\kappa)\mu_1+\kappa\mu_2)^2 \tag{15}$$

We expand the parentheses in (15) and rearrange the terms so that they form a second order polynomial in κ:

$$\begin{gathered}
(\sigma_1{}^2 + \mu_1^2 - \kappa\sigma_1{}^2 - \kappa\mu_1{}^2) + (\kappa\sigma_2{}^2 + \kappa\mu_2{}^2) - \\
(\kappa^2\mu_1^2 + 2\kappa\mu_1\mu_2 - 2\kappa^2\mu_1\mu_2 + \mu_1^2 - 2\kappa\mu_1^2 + \kappa^2\mu_2^2) = \\
\left(-\mu_1{}^2 + 2\mu_1\mu_2 - \mu_2{}^2\right)\kappa^2 \\
+\left(-\sigma_1{}^2 + \mu_1{}^2 + \sigma_2{}^2 + \mu_2{}^2 - 2\mu_1\mu_2\right)\kappa \\
+\sigma_1{}^2 = \\
-\left(\mu_1{}^2 - 2\mu_1\mu_2 + \mu_2{}^2\right)\kappa^2 \\
+\left(\left(\mu_1{}^2 - 2\mu_1\mu_2 + \mu_2{}^2\right) - \left(\sigma_1{}^2 - \sigma_2{}^2\right)\right)\kappa \\
+\sigma_1{}^2 = \\
\boxed{-(\mu_1-\mu_2)^2\kappa^2 + \left((\mu_1-\mu_2)^2 - (\sigma_1^2-\sigma_2{}^2)\right)\kappa + \sigma_1{}^2}
\end{gathered} \tag{16}$$

This concludes the derivation of equation (2) in Chapter 3.

Chapter 8

Paper IV: Seismic Volume Visualization for Horizon Extraction

Seismic Volume Visualization for Horizon Extraction

Daniel Patel[*]
Christian Michelsen Research, Bergen, Norway

Stefan Bruckner[†]
Institute of Computer Graphics and Algorithms, Vienna University of Technology, Austria

Ivan Viola [‡]
Department of Informatics, University of Bergen, Norway

Eduard M. Gröller [§]
Institute of Computer Graphics and Algorithms, Vienna University of Technology, Austria

Abstract— Seismic horizons indicate change in rock properties and are central in geoscience interpretation. Traditional interpretation systems involve time consuming manual volumetric seeding for horizon growing. We present a novel system for rapid interpreting and visualizing seismic volumetric data. First we extract horizon patches by preprocessing the seismic data. Then during interaction the user can in realtime assemble the horizon patches into horizons. Traditional interpretation systems use gradient-based illumination models in rendering of the seismic volume and polygon rendering of horizon surfaces. We employ realtime gradient-free forward-scattering rendering of seismic volumes yielding results similar to high-quality global illumination. We use an implicit surface representation of horizons allowing for a seamless integration of horizon rendering and volume rendering. We present a collection of novel techniques constituting an interpretation and visualization system highly tailored to seismic data interpretation.

Index Terms—Seismic interpretation, Seismic horizons, Volume rendering, Ambient occlusion, GPU acceleration.

1 Introduction

Energy is one of the most important resources in today's societies. Over half of the world-energy needs come from oil and gas [14]. This results in high demands for hydrocarbon resources and makes their identification and extraction economically valuable. In this paper we present a system for rapid seismic interpretation. Hydrocarbons are found in subsurface reservoirs. To identify them, subsurface volumetric data is obtained by sending sound waves into the ground and receiving and processing their echoes. The resulting seismic reflection volume then goes through a complex and time consuming manual interpretation for identifying subsurface structures which may hint to where hydrocarbons are trapped.

The Earth subsurface consists of material layers with distinct mineral densities and porosity characteristics. The interfaces between these layers are called horizons, they are seen as bright or dark lines in gray-level reflection data and are central structures for interpretation. Other structures such as faults, channels, salt bodies, and gas chimneys are mainly identifiable by their interaction with horizons. Faults are generally sub-vertical fractures which have shifted horizons on either sides; they are thus identified as discontinuity in horizons. Salt bodies are homogeneous units of salt. Due to the high seismic wave velocity of homogeneous salt, such structures can have strong reflections at their boundaries and low or no internal reflections. Areas void of horizons can thus indicate the presence of salt bodies having the property of shadowing the underlying seismic. Subsurface leakages of gas, called gas chimneys, can be indicated by the up-bulging of horizons around them and the fragmentation of the horizons in their path. A reservoir in a porous rock formation can be accurately pinpointed by identifying it's upper and lower boundary. The different rock materials at the boundaries give rise to horizons in the reflection data. Therefore horizons can be central in delineating reservoirs.

In addition to these descriptive properties of horizons, horizons are perhaps the most easily identifiable objects in the seismic volume and thus of the most important objects during interpretation. As opposed to most other seismic structures, horizons can directly be identified with image processing techniques such as ridge and valley detection. For these reasons we focus on horizon identification in this work. We present an approach for automatically identifying horizons and visualizing them.

Visualizing 3D seismic data is challenging due to the the dense nature of the data, the high presence of noise, and the difficulty to identify, manipulate and select structures in 3D. These problems have the consequence that the interpretation of 3D seismic reflection data is nowadays carried out as lengthy series of interpretations of parallel 2D seismic cross-sections throughout the 3D volume. In this paper we present a novel approach that enables rapid interpretation using interactive computer-assisted 3D visualization. The basic idea is to precompute horizon candidates from the dataset and partition them into small "building blocks". The building blocks are then selected by the geoscientist during a rapid interpretation process and assembled together into correct horizons.

As opposed to the accurate, qualitative and slow interpretation process currently used, our is quick, quantitative and sub-accurate. It enables an early overview of the data so one can identify where to perform accurate interpretation later. The currently used interpretation workflows do not support getting overviews. We propose a 3D rapid interpretation process, similar to the trend seen in architectural CAD systems, where 3D sketchy systems [11] have become superior during idea prototyping over traditional 2D floor-based modeling software [2]. For detailed interpretation and corrections, interpreted horizons can later be imported into existing systems. In more detail-oriented systems, the qualitative and highly user steered tasks of identifying horizon-splitting structures such as faults and unconformities can be performed. These systems can then split horizons and perform manipulation beyond the resolution of the smallest building blocks presented here.

A major challenge for providing useful 3D interactive visualization, is the choice of an appropriate 3D rendering algorithm. Gradient-based shading [18], is effective for depicting volumetric data having clear material boundaries. The gradient vectors are then used as normals in the Phong illumination model [25]. Gradient-based methods, whether based on central differences or more complex seismic dip and azimuth estimations [20], are in general sensitive to high-frequency noise. As acquisition is based on acoustics, seismic data are typically of noisy nature which lacks distinct material boundaries. Gradient-based shading on seismic data thus introduces distracting artifacts which makes interpreting seismic data 3D renderings difficult. Other approaches, such as unshaded direct volume rendering tend to depict seismic data as a homogeneous cloud without distinct features. Thus, common approaches are frequently unsuitable for visualizing seismic data. In this paper we present a gradient-free ambient-occlusion like method for shaded volume rendering of seismic data that address these issues.

Related work is reviewed in Section 2. The high-level concept is

[*]e-mail:daniel@cmr.no
[†]e-mail:bruckner@cg.tuwien.ac.at
[‡]e-mail:Ivan.Viola@ii.uib.no
[§]e-mail:groeller@cg.tuwien.ac.at

described in Section 3. Section 4 and 5 describe the horizon extraction and visualization stage. Demonstration of the proposed technology is presented in Section 6 and conclusions are drawn in Section 7.

2 Related Work

Several research works and commercial solutions present interpretation and visualization algorithms for 3D seismic data. Horizon-interpretation algorithms are typically semi-automatic and require a detailed and time consuming user involvement. Pepper and Bejarano [24] give an overview of computer-assisted interpretation algorithms for seismic data. Commercial software used in oil companies include HydroVR [19] and Petrel [31]. Techniques that perform horizon extraction usually take a semi-interactive approach. The user manually places a seed point on a horizon in a seismic slice and adjusts growing parameters before starting a growing process of the horizon. Several growing algorithms exist. Castanie et al. [7] propose user seeding followed by growing based on the local waveform of the seedpoint. The local waveform is defined as the vertical 1D signal one gets from the seismic values in a local neighborhood above and below a sample. Interpretation software [31] performs growing in voxels that have been thresholded or in extrema or zero crossings of the local waveform.

These methods are not fully interactive due to the need to wait for the growing to finish. To our knowledge there are no commercial solutions that support fully interactive and thereby rapid horizon interpretation of seismic data. For generality, the interpretation software often presents the user many parameters to specify for the growing process. Typically the user changes parameters and re-grows until a satisfactory result is obtained. The parameters to be set are related to the internal growing/image processing algorithms, such as the topology of the connectivity neighborhood to use, and can be difficult to understand for an average seismic interpreter. One might argue that too much low level control is given to the user. In our system we aim at avoiding the need for parameter tweaking.

On the other hand, there are completely automatic approaches for horizon interpretation [4]. In this work, voxels are mapped to points in an n-dimensional space based on their local waveforms. Clusters are then identified in the n-dimensional point cloud. Probably due to the low control and predictability of the outcome, long cycles of setting parameters and waiting periods before the results are available, such methods have not gained popularity.

Automatic horizon interpretation as preprocessing for rapid interpretation of 2D seismic slices has been proposed by Patel et al. [22]. An extension of their technique into 3D would be problematic as their horizon analysis produces a set of line segments for each 2D seismic slice. A 3D horizon preprocessing method was presented in the work by Faraklioti and Petrou [9]. However their connected component analysis was only able to grow out planar and well-defined horizons.

In our work we discuss the concept of rapid horizon interpretation by focusing on interpretation of seismic horizons, high quality 3D visualization and quick interaction techniques. We automatically extract surfaces that with high likelihood coincide with horizons. We subdivide these surfaces into smaller patches using an existing mesh clustering algorithm [1]. An overview of mesh clustering algorithms can be found in the work by Shamir [33]. Shamir categorizes clustering methods into greedy quick methods, global slow methods, growing methods, hierarchical methods and spectral analysis methods. We chose a hierarchical greedy method due to the generation of hierarchy information which we use during interpretation. Also the method does not require to define initial seed faces which affects the resulting subdivision. For a seamless integration of horizon-patch visualization with seismic-volume visualization, we represent the patches through a distance volume with segmentation masks. Existing distance transform techniques have been surveyed by Jones et al. [15]. Since we perform the distance transform in a preprocessing step, we are not dependent on speed. We therefore create a computationally expensive but analytically accurate distance transform by in essence considering the analytical distance to all triangles from each voxel [16].

The basic concept of our approach is that the horizon interpretation is carried out directly in 3D. Several aspects for the 3D visualization of seismic volumes have been investigated in earlier works. Plate et al. [27, 26] and Castanie et al. [7] discuss handling multiple large seismic volumes in the rendering pipeline. Ropinski et al. [30] discuss volume rendering of seismic data in VR with volumetric cutouts. Illustrative rendering techniques employing textures have been proposed for presentation of interpreted seismic data [23].

Almost all papers use gradient-based illumination to render seismic volumes. Most calculate the gradient directly from the data. One exception is the work by Silva et al. [34] where some consideration has been taken to adapt the illumination model for seismic data. In their work the gradients are calculated from a derived seismic phase volume which in some cases gives better results. The rendering approaches currently used for seismic data, are not suitable for horizon display. With a gradient-based illumination model it is very difficult to identify subtle horizon structures because of a lack of depth cues and a strong visual dominance of noise.

One way to add depth cues to volume rendered images is to use a more realistic illumination model. Yagel et al. [36] employ recursive ray-tracing which allows effects such as specular reflection and shadows. Behrens and Ratering [3] add shadows to texture-based volume rendering. The model presented by Kniss et al. [17] captures volumetric light attenuation effects including volumetric shadows, phase functions, forward scattering, and chromatic attenuation. Rezk-Salama [28] presents an approach for GPU-based Monte Carlo raytracing. Max [21] gives a comprehensive overview of different optical models for volume rendering. A problem of increasing the physical realism is, however, that these models often lack control over the specific appearance of certain structures of interest. As they are based on actual physical laws, it is difficult to control individual visualization properties separately.

Other approaches employ visually plausible approximations of realistic lighting effects. Stewart [35] introduces vicinity shading, a view-independent model to enhance perception of volume data. It is based on occlusions in the local vicinity of a sample point resulting in shadows in depressions and crevices. In a similar approach, Hernell et al. [13] use a local approximation of ambient occlusion. Another step towards real-time global illumination for volumes has been done by Hernell et al. by local piecewise integration [12]. Desgranges et al. [8] use incremental blurring to achieve shading effects without the use of a gradient. The approach by Bruckner and Gröller [6] is able to generate shadowing and emission effects in a view-dependent manner while still allowing interactive transfer-function modifications. Our pipeline uses a gradient-free illumination model to visually reduce the noise and to provide depth cues.

3 Rapid horizon-interpretation pipeline

The pipeline depicted in Figure 1 is conceptually divided into two parts. The first part is a preprocessing step that extracts horizon candidates in advance of the visualization so the user neither must set parameters for the horizon tracking nor wait for the results. This is covered in Section 4. The second part deals with expressive visualization and interaction and is covered in Section 5.

4 Horizon Extraction

Performing automatic feature extraction on seismic data is problematic due to the noisy nature of seismic data. Even manual feature extraction is difficult and involves some educated guessing. We design our horizon extraction with this in mind by providing the user with a large collection of precalculated horizon patches. A set of plausible horizon patches based on a horizon-growing and a surface subdivision algorithm is precalculated and presented to the user to pick from. The user can then build horizons by assembling these patches.

We have tried to optimize the balance of work between the user and the computer to enable rapid horizon extraction. We tried to make a balance so the computer is not assisting too little and requiring unnecessary user input or too much by inflexibly dictating the horizon shapes. With our approach, the user is freed from being constantly interrupted during interpretation by having to explicitly seed horizons

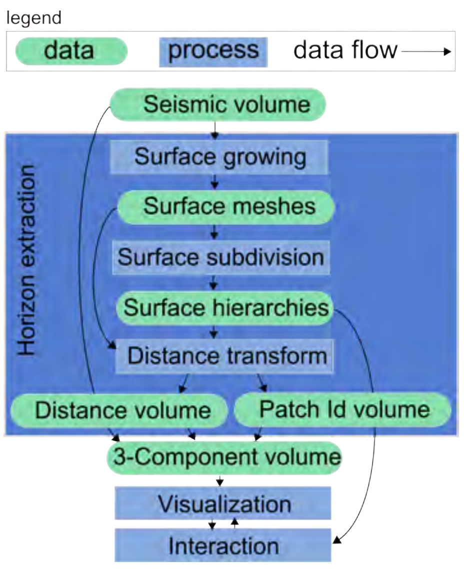


Fig. 1. An overview of the data and the processing in our pipeline. All preprocessing is performed in the big blue box vertically labeled 'Horizon extraction', followed by visualization and interaction for assembling correct horizons. The horizon extraction is further described in Figure 2 and Figure 4.

followed by waiting for the horizons to grow. Explicit method have advantages when it comes to fine-tuning the horizons. However we focus on rapid interpretation for getting a first overview of the data. This overview can complement existing workflows so later, detailed interpretation can be focused on areas that have been identified as important with our method.

We enable the user to choose which level of horizon granularity she wishes to operate on and can for each picked horizon patch change the granularity level. For this to work we first grow all possible horizons in the volume and then we perform a hierarchical subdivision of the horizons into smaller patches. The hierarchy is stored in a tree structure with the smallest patches as leaf nodes. The user can navigate in the hierarchy and work with patches of the size and shape which reflect the interpretation best.

As described in Section 2, several horizon identification methods exist. Methods that do not rely on growing, such as performing a clustering of the voxels based on their local neighborhood, result in segmentation masks with no connectivity information. Growing methods are topology aware. This allows for constrained growing and result in connectivity information that can be used to polygonize the horizons.

We seed and grow each voxel in the volume. Growing is only performed in voxels that are on positive extremas of the vertical waveform, i.e. voxels that have a lower valued voxel directly above and below it. This gives a robust tracing of horizons of varying intensities and ensures that a sheet topology is achieved. Voxels are visited only once to avoid intersecting and overlapping surfaces. An example of a growing originating in one voxel is seen in Figure 2. The width of each growing frontier, defined as the number of neighboring points constituting it, must be above a certain size or else the growing of that border is terminated. The larger the minimum width is, the less bifurcations in the surface is created. For instance, a minimum width of 1 will create non-sheet surfaces as surfaces will branch out in lines which in turn may become surfaces. The width parameter heuristically controls the minimum radius a surface can have. We achieved a good balance between missing out on horizons and over-segmentation by using a width of 4 voxels. To create surfaces, we triangulate the frontier during growing. We achieve this by triangulating the strips consisting of the points in the current borders (green points in Figure 2) and the points in the borders from the previous step (red points in Figure 2). The grown surface is represented as a vertex list of the coordinates of the grown voxels followed by an index list of triangle 3-tuples with indices to these vertices.

Frequently during surface growing, what an interpreter would consider as separate horizons, are growing into a single surface. This typically happens for horizons close to each other. Due to noise or unconformities, a growing frontier might jump to the horizon directly above or below. It will then start growing on that horizon while other frontiers continue on the original horizon. Erroneous growings result in surfaces consisting of multiple merged horizons instead of isolated correct ones. To address this, we subdivide each grown surface into smaller pieces for the user to select. The user can in realtime pick such patches, subdivide them further if necessary and thereby assemble correct horizon surfaces.

For subdividing the triangulated surfaces into smaller pieces we are using the method described by Attene et al. [1] that builds on the work by Garland et al. [10]. This greedy method creates a hierarchical surface segmentation by starting with single-triangle surfaces and iteratively merging neighboring surfaces into larger ones by choosing the neighbors that maximize the flatness of the joined surface. Details can be read in the original paper [1]. This process continues until all surfaces are merged into one. See Figure 3 for an illustration of this process. The merging operations can be described as the interior nodes of by a binary tree having single triangles as leaf nodes. We create and store this binary surface tree for each grown surface. The binary tree is used later during interpretation. It enables the user to work with subsurfaces at a user defined granularity level by moving up and down in the hierarchy. After selecting a leaf node, the user can navigate up in the tree and select surfaces of appropriate sizes for a quick horizon assembly.

The subdivisions of a binary surface tree is depicted in Figure 3. For each consecutive image, a jump of 4 levels up in the tree is performed. Distinct colors represent distinct nodes in the corresponding tree level. In the top image, the tree leaves are seen. Each color represents a single triangle of the surface. However, single triangles have approximately the size of a voxel and are prohibitively small to work on. Therefore we prune the tree so that during interaction, the child nodes are not single triangles, but groups of triangles of a predefined minimum size. We chose to prune the tree into having leaf nodes with minimum 500 triangles. Each leaf node consists of a unique leaf-id followed by a list of indices to the triangle 3-tuples of the surface as described earlier. Each interior node consists of pointers to two child nodes and the number of triangles in its subtree.

For an integrated representation of seismic data and the surface tree, we perform a distance transform on all leaf surfaces which together constitute all grown surfaces. For each voxel in the seismic volume we calculate and store the distance to the closest point on a leaf surface together with the unique id of that leaf surface. The voxel's leaf id creates a link from the volume to the leaf node in the surface tree which represents the closest leaf surface to the picked voxel.

This link enables 3D picking of horizons followed by a hierarchical navigation in the surface tree from the leaf node to larger horizon surfaces. We merge the seismic volume, the distance field volume and the leaf surface id into one 3-component volume. This enables quick lookups during interaction. See Figure 4 for an illustration of the distance component. The distance component combined with the leaf-id component allows for several volume rendering modes. Picked surface leaf id's and the distance volume constitute a segmentation mask around a horizon. This segmentation mask can be used to render the implicit horizon surface with a user defined color and thickness. Alternatively in an inverse mode, seismic rendering is only performed in the segmentation mask with full transparency outside. If instead opaque rendering is performed in the segmentation mask then the seismic data is projected on the horizon at a user defined distance from the horizon.

5 Volume and Horizon Visualization

During interpretation, high quality volumetric rendering is performed, horizons are visualized and the user can interact with the data. These three topics are covered respectively in the three following subsections.

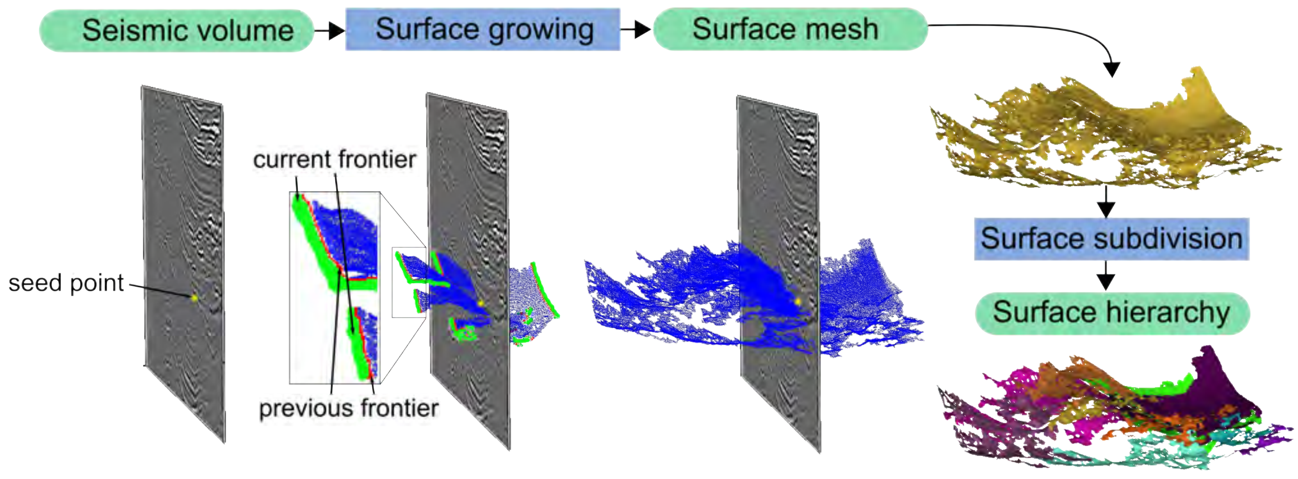


Fig. 2. The process of growing from one specific seedpoint (yellow point). The green borders in figure 2 show the growing frontier and the red show the previous frontier.

5.1 Gradient-free volume visualization

Surface-based shading using the gradient direction as a substitute for the surface normal is most effective when there are clearly identifiable material boundaries. An example is medical volume data acquired by computed tomography (CT). For seismic data, however, this approach commonly results in noisy images which are difficult to interpret. Unshaded visualization, on the other hand, suffers from a lack of clearly delineated features. In our approach we want to visualize the extracted horizon information in conjunction with the original seismic data to enable verification of the extraction process. Thus, a volume visualization technique is required which gives a good indication of the overall structure of the data set.

One potential candidate to fulfill these requirement is ambient occlusion. The incoming radiance at a point is determined by shooting rays in all directions from a point and averaging its degree of occlusion by other parts of the volume. The result is an approximation of global diffuse illumination. It produces soft shadowing effects which give a good indication of spatial relationships. In volume visualization the opacity at any point is determined by the transfer function. Ambient occlusion therefore requires an expensive computation step every time the transfer function is modified. The approach of Ropinski et al. [29] relies on local histogram clustering to precompute this information for all possible transfer function settings. The precomputation is extremely time and memory consuming, in addition, the high frequency and noisy seismic data makes a bad candidate for local histogram clustering. We therefore choose to employ a view-dependent method which generates similar shadowing effects but still allows interactive transfer function manipulation.

Inspired by the approach of Bruckner et al. [6], Desgranges et al. [8] as well as recent work by Schott et al. [32] we use a slice-based volume rendering method where an incremental blurring operation is applied to the accumulated opacity buffer. This buffer is then used to determine the degree of shadowing applied to subsequent slices. The volume is traversed in front-to-back order using view-aligned slicing. Two buffers C_i, the color buffer, and O_i, the occlusion buffer, are written for each slice i. For each sample on the slice, we apply a Gaussian low-pass filter G to the occlusion buffer O_{i-1} of the previous slice and combine it with the opacity contribution α_i of the current slice using additive blending:

$$O_i = G * O_{i-1} + \alpha_i$$

Based on this incremental filter operation, for each sample on the slice a modulation factor λ_i is computed which determines the degree of visibility:

$$\lambda_i = \frac{1}{1 + G * O_{i-1}}$$

The color contribution c_i is then multiplied by this factor and c_i and α_i are used for conventional alpha-blending using the over operator to generate the color buffer C_i.

This simple approach generates soft shadowing effects akin to ambient occlusion but can be performed interactively on current graphics hardware. Due to the incremental filtering of the occlusion buffer, darkening from a highly occlusive region will reduce smoothly with distance leading to a visual appearance as if the volume was illuminated by a large area light source located at the camera position. By setting ambient values in the occlusion buffer for certain objects in the volume one achieves the effect of having objects cast light instead of shadows. Examples of such light emitting objects will be shown in the subsequent section.

In Figure 5 we give a comparison between a Phong shaded seismic volume and our rendering approach. As can be seen, our rendering is less susceptible to high frequency noise and gives a better depth perception of the data. Both images use the same transfer function for comparison reasons with a simple black to white color ramp. The lower values of the reflection data are set to transparent while the higher values are set to be opaque.

5.2 Horizon Visualization

We visualize the horizons together with the seismic data. The horizons are represented implicitly by the distance component and the patch-id component. As opposed to a geometric representation, the implicit representation enables a seamless co-rendering of horizons and seismic data in one rendering pass. A mapping from patch-ids to RGBA values defines the color and opacity of each patch in the volume. Initially all horizon patches are set to fully transparent.

During the slice-based volume rendering, the distance and patch-id component of the pixels of each slice that is composited, is looked up in the volume. If the distance is less than a user defined value, the pixel is modulated with the color and transparency assigned to the patch id. Thus the mapping defines the selection and appearance of horizons. Simple variations of the colored horizon rendering can easily be achieved. Volume rendering of the seismic data can be restricted to only take place in the vicinity of selected horizons and having transparency everywhere else.

Other seismic attributes from the same seismic study, such as a fault probability volume, can be opaquely projected on the horizons. This would express the fault probability at any point on a horizon surface.

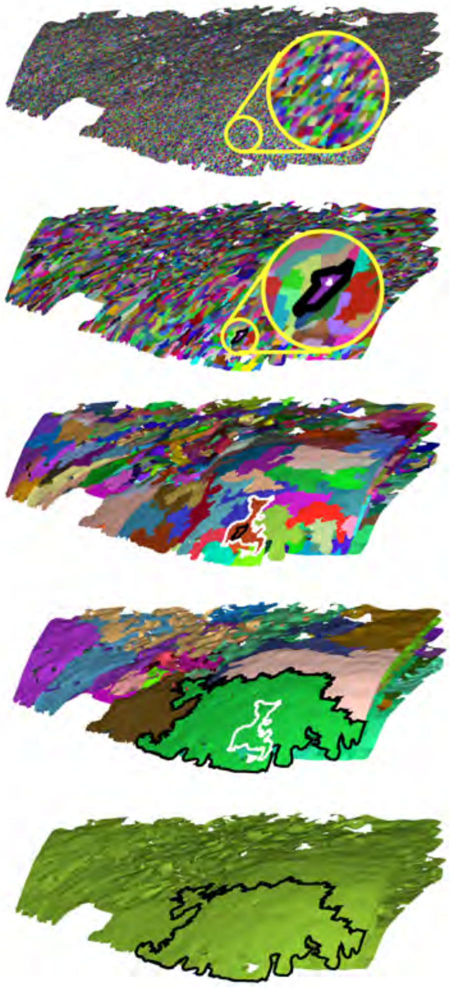


Fig. 3. Different levels of a surface hierarchy made from a grown surface. The top image shows the initial one-triangle surfaces. Each successive image shows the resulting merging after four merge levels in the binary hierarchy tree. One triangle is magnified in white at the top image and its successive higher level surfaces are shown delineated with alternating white and black borders.

Advanced light effects can be achieved with the incremental filtering used for creating shadows during volume rendering. For instance, instead of casting shadows, we can let the horizons cast their colors, thereby simulating emissive glowing horizons. The glowing effect can be used to expressively highlight important horizons embedded in the seismic volume or to distinguish them from other horizons.

5.3 Interaction

We designed the system with focus on simple interaction that supports easy and rapid interpretation of horizons. Intuitive interaction with the seismic volume is achieved by using the patch-id component. This gives the ability to select leaf-patches by clicking directly on the volume rendered image. When a leaf-patch is selected, its surface hierarchy is available and can be navigated in. This allows the user to iteratively select higher level surfaces until the optimal size is found.

Volumetric horizon selection is achieved by first storing the 2D screen-space coordinate of the point the user selects. While performing front-to-back compositing of volume slices during volume rendering, the first hit with a nontransparent voxel is detected at the 2D coordinate on the compositing slice. The corresponding patch-id is fetched from the patch-id volume. This id identifies the selected leaf patch and its color mapping is set from full transparency to a color indicating it is selected.

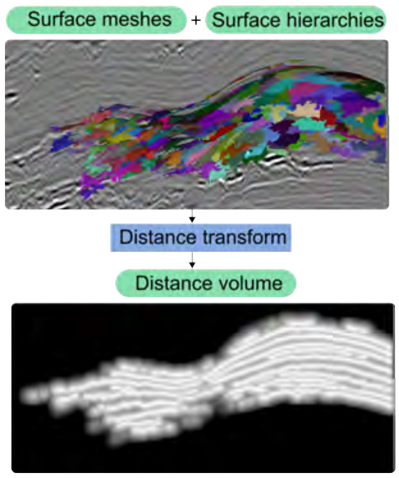


Fig. 4. The distance transform. One grown surface mesh is shown in 3D at the top intersecting a slice of the seismic data. The colors represent patches at the lowest level in the pruned hierarchy. The bottom image shows a slice through the resulting distance transform with white representing distance zero and darker colors representing higher distances.

Due to the dense nature of seismic data, we have defined a roaming box which confines the volume rendering. The user can easily move the roaming box through the volume and select horizons by clicking on the sides of the roaming box. Section 6 describes such an interaction scenario.

6 Results

In this section we present a scenario of navigating and interacting for rapid horizon interpretation. In the scenario, two horizons are interpreted by using patch-picking and hierarchical composition of patches into larger surfaces. A monotonous white transfer function with low to high values mapped linearly from transparent to opaque. Dark areas are due to shadows cast by opaque horizons through transparent space onto other opaque horizons. Typically transfer functions for seismic-volume rendering use a black to white gradient to easier discern horizons. With shadowing this is not necessary. This simplifies the transfer function setup supporting rapid interpretation.

To identify a horizon for interpretation the side of the roaming box is moved through the volume. An interesting horizon is seen on the frontside of the roaming box (Figure 6a) and the horizon is selected using point-and-click (red arrow in zoom-in of Figure 6a). This results in a selected leaf-patch shown in pink. Instead of continuing to select small patches along the horizon, the hierarchy information is utilized. By using the keyboard up-arrow all patches in the hierarchy level above the current one is colored pink. The user can navigate up to the highest hierarchy level that yields a correct horizon and use that as a starting point of the horizon assembly. Figures 6b-e show increasing surface hierarchy levels. Going from the hierarchy level in Figure 6e and one level up shown in Figure 6f, erroneous horizons above the current are included. The error is revealed by the irregular and overlapping structures that do not exist in horizons. Due to the cast shadows provided by our method such erroneous cases are easy to identify. Therefore the user goes back to the previous level shown in Figure 6e and uses this as a starting point. The user has now selected the horizon at the highest level in the hierarchy without errors. This surface constitutes a horizon with holes. The holes might consist of unselected horizon patches. The user therefore attempts to fill these holes by selecting potentially segmented patches located in the holes. To select these patches, the roaming box side is moved to intersect the holes (Figure 7a). The red arrow in the zoom-in show where the user selects the horizons. In Figure 7b the resulting selected patch is shown in green. The process of moving the roaming box side over the horizon and filling out holes where possible is performed over the entire horizon. The result can be seen in Figure 7c with green patches representing filled-in holes. Now one horizon has been interpreted. The user selects another horizon color (blue) and repeats the procedure. Two interpreted horizons can be seen in 7e. Different rendering types are shown in 7f-k. In 7f is shown the top horizon semitransparently so the underlying data can be seen. In 7g is shown the top horizon with an emissive factor. Emissiveness is good for bringing attention to a certain horizon. Figure 7h shows the bottom horizon emissively. In 7i is shown both horizons emissively. On the side face of the roaming box one can see how the emissiveness interacts with the volume rendering by casting light from the emissive object onto its surroundings. Emissiveness can be used to easier identify interpreted horizons embedded inside a volume rendering. This is due to their prominent visualization. Several other rendering techniques exist for giving prominence to selected objects but with our rendering method the prominence is achieved using inherent and natural properties of the lighting model. In 7j is shown the horizons extruding from an opaque cube. The implicitly represented surfaces together with ambient occlusion-like shading gives this rendering comparable quality with a hand a drawn illustration. In 7k is shown a scenario with seismic data shown around a hypothetical boring well and its relation to the two horizons. The interpretation just presented could easily be performed in less than ten minutes. This short interpretation time underlines the potential of our approach.

6.1 Implementation details

The software has been developed using the VolumeShop [5] framework. For the preprocessing, about 2000 surfaces were grown and separately subdivided, constituting altogether about 2 million triangles. The hierarchies for each surface is stored in memory in a downstripped version containing only patch-id's in the leaf nodes. The size is then less than 1MB. The original hierarchies and the surfaces are stored on file. This enables exporting the horizons into commercial seismic interpretation systems for further processing. Implicitly representing the surfaces removes the need to store and maintain visualization properties of the surface geometries in memory and enables a single-pass integrated rendering, however with the penalty of increased volume size.

With unoptimized preprocessing, it took 1 hour to calculate the distance transform using a maximal distance of 10 voxels. The unoptimized brute force surface growing from each voxel in the volume took 5 hours and it took 1 hour for the hierarchy creation of the grown surfaces. We ran the preprocessing on an AMD Athlon 64, Dual core processor 3800, 1.8 Ghz. For rendering we used an Nvidia GeForce 275. Our dataset was of size 256x256x256. One byte was used to represent the seismic value, one byte for the distance and two bytes for the patch id. This resulted in a 3-component volume of size 64MB thus 4 times larger than the original seismic volume. During interaction we achieved 25fps in a 500 by 500 window. The high-resolution images of size 1000x1000 used in this article were taken with 8 samples per voxel resulting in a speed of 7-14 fps.

7 Conclusion and Future Work

We have presented a system for rapid interpretation and expressive visualization of seismic horizons by carefully combining appropriate technologies. Performing seismic preprocessing before interpretation, combining the fields of seismic horizon growing with surface segmentation and representing horizons implicitly instead of as triangle meshes enabling a one-pass rendering has to our knowledge not been applied on seismic data. Neither has the illumination model for clearer visualization of noisy acoustic seismic reflectance data.

Future work includes investigating the emission of transparency instead of shadow or color, from horizons, so selected horizons can stand out from the surrounding volume by suppressing the data around it. Integrating our system into existing interpretation systems for a more seamless move from rapid quantitative interpretation to detailed qualitative interpretation is also of interest.

8 Acknowledgements

Data used in this article is by courtesy of Norske ConocoPhillips AS, ExxonMobil Norway AS, Total E&P Norge AS, StatoilHydro AS and Petoro AS.

References

[1] M. Attene, M. Spagnuolo, and B. Falcidieno. Hierarchical mesh segmentation based on fitting primitives. *The Visual Computer*, 22(3), sep. 2003.

[2] Autodesk AutoCAD, http://www.autodesk.com/, 2009.

[3] U. Behrens and R. Ratering. Adding shadows to a texture-based volume renderer. In *Proceedings of IEEE Symposium on Volume Visualization 1998*, pages 39–46, 1998.

[4] H. Borgos, O. Gramstad, G. V. Dahl, P. L. Guern, and L. Sonneland. Extracting horizon patches and geo-bodies from 3d seismic waveform sequences. *SEG/New Orleans 2006 Annual Meeting*, pages 1595–1599, 2006.

[5] S. Bruckner and M. E. Gröller. Volumeshop: An interactive system for direct volume illustration. In H. R. C. T. Silva, E. Gröller, editor, *Proceedings of IEEE Visualization 2005*, pages 671–678, Oct. 2005.

[6] S. Bruckner and M. E. Gröller. Enhancing depth-perception with flexible volumetric halos. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1344–1351, 2007.

[7] L. Castanie, B. Levy, and F. Bosquet. Volumeexplorer: Roaming large volumes to couple visualization and data processing for oil and gas exploration. *Proc. of IEEE Visualization '05*, pages 247–254, 2005.

[8] P. Desgranges, K. Engel, and G. Paladini. Gradient-free shading: A new method for realistic interactive volume rendering. In *Proceedings of Vision, Modeling, and Visualization 2005*, pages 209–216, 2005.

[9] M. Faraklioti and M. Petrou. Horizon picking in 3D seismic data volumes. *Machine Vision and Applications*, 15(4):216–219, 2004.

[10] M. Garland, A. Willmott, and P. S. Heckbert. Hierarchical face clustering on polygonal surfaces. In *I3D '01: Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 49–58, New York, NY, USA, 2001. ACM.

[11] Google SketchUp, http://sketchup.google.com/, 2009.

[12] F. Hernell, P. Ljung, and A. Ynnerman. Interactive global light propagation in direct volume rendering using local piecewise integration. In *Proceedings of IEEE/EG International Symposium on Volume and Point-Based Graphics*, pages 105–112, 2008.

[13] F. Hernell, A. Ynnerman, and P. Ljung. Efficient ambient and emissive tissue illumination using local occlusion in multiresolution volume rendering. In *Proceedings of Volume Graphics 2007*, pages 1–8, 2007.

[14] A. Iske and T. Randen, editors. *Atlas of 3D Seismic Attributes, Mathematics in Industry, Mathematical Methods and Modelling in Hydrocarbon Exploration and Production*. Springer, Berlin Heidelberg, 2006.

[15] M. Jones, J. Baerentzen, and M. Sramek. 3d distance fields: a survey of techniques and applications. *IEEE Transactions on Visualization and Computer Graphics (IEEE TVCG)*, 12(4):581–599, July-Aug. 2006.

[16] M. W. Jones. 3d distance from a point to a triangle. *Technical Report CSR-5-95, Department of Computer Science, University of Wales Swansea*, February 1995.

[17] J. Kniss, S. Premoze, C. Hansen, P. Shirley, and A. McPherson. A model for volume lighting and modeling. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):150–162, 2003.

[18] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8:29–37, 1987.

[19] E. M. Lidal, T. Langeland, C. Giertsen, J. Grimsgaard, and R. Helland. A decade of increased oil recovery in virtual reality. *IEEE Computer Graphics and Applications*, 27(6):94–97, 2007.

[20] K. Marfurt. Robust estimates of 3D reflector dip and azimuth. *Geophysics*, 71:P29, 2006.

[21] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

[22] D. Patel, C. Giertsen, J. Thurmond, J. Gjelberg, and M. E. Gröller. The seismic analyzer: Interpreting and illustrating 2d seismic data. *IEEE Transactions on Visualization and Computer Graphics (IEEE TVCG)*, 14(6):1571–1578, Oct 2008.

[23] D. Patel, C. Giertsen, J. Thurmond, and M. E. Gröller. Illustrative rendering of seismic data. In *Proceedings of Vision Modeling and Visualization*, pages 13–22, 2007.

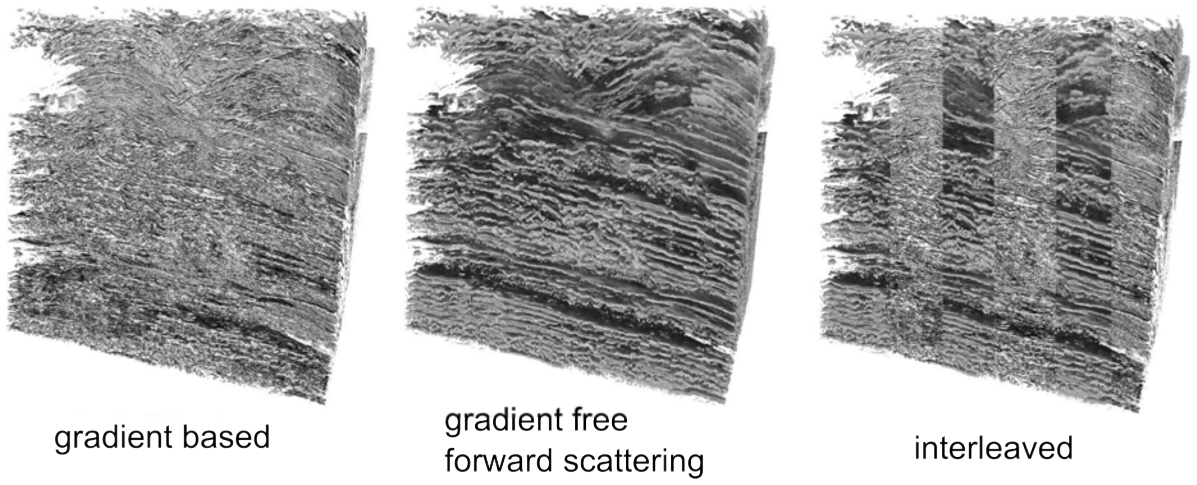


Fig. 5. A comparison of gradient (left) and gradient-free shading (middle). In the right image, the two left images are interleaved and one can see how the depth perception is improved and high frequency noise reduced in the gradient-free method.

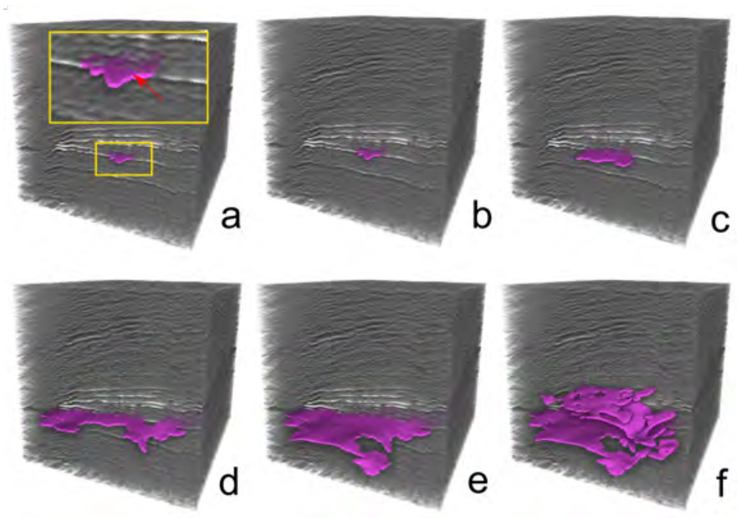


Fig. 6. Selecting a leaf-patch (a) and navigating stepwise up in the hierarchy (b-f) until an erroneous surface is shown (f).

[24] R. Pepper and G. Bejarano. Advances in seismic fault interpretation automation. In *Search and Discovery Article 40170, Poster presentation at AAPG Annual Convention*, pages 19–22, 2005.

[25] B. T. Phong. Illumination for computer generated pictures. *ACM Communications*, 18(6):311–317, 1975.

[26] J. Plate, T. Holtkaemper, and B. Froehlich. A flexible multi-volume shader framework for arbitrarily intersecting multi-resolution datasets. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1584–1591, 2007.

[27] J. Plate, M. Tirtasana, R. Carmona, and B. Fröhlich. Octreemizer: a hierarchical approach for interactive roaming through very large volumes. *Proc. of VISSYM '02*, pages 53–64, 2002.

[28] C. Rezk-Salama. Gpu-based monte-carlo volume raycasting. In *Proceedings of Pacific Graphics 2007*, pages 411–414, 2007.

[29] T. Ropinski, J. Meyer-Spradow, S. Diepenbrock, J. Mensmann, and K. H. Hinrichs. Interactive volume rendering with dynamic ambient occlusion and color bleeding. *Computer Graphics Forum*, 27(2):567–576, 2008.

[30] T. Ropinski, F. Steinicke, and K. H. Hinrichs. Visual exploration of seismic volume datasets. *Journal Proc. of WSCG '06*, 14:73–80, 2006.

[31] Schlumberger and Petrel. Petrel seismic interpretation software, schlumberger information solutions (sis).

[32] M. Schott, V. Pegoraro, C. Hansen, K. Boulanger, J. Stratton, and K. Bouatouch. A directional occlusion shading model for interactive direct volume rendering. *Computer Graphics Forum*, 28(3):855–862, june 2009.

[33] A. Shamir. A survey on mesh segmentation techniques. *Computer Graphics Forum*, 27(6):1539–1556, 2008.

[34] P. M. Silva, M. Machado, and M. Gattass. 3d seismic volume rendering. *Eighth International Congress of The Brazilian Geophysical Society*, 22(3):181–193, 2003.

[35] A. J. Stewart. Vicinity shading for enhanced perception of volumetric data. In *Proceedings of IEEE Visualization 2003*, pages 355–362, 2003.

[36] R. Yagel, A. Kaufman, and Q. Zhang. Realistic volume imaging. In *Proceedings of IEEE Visualization 1991*, pages 226–231, 1991.

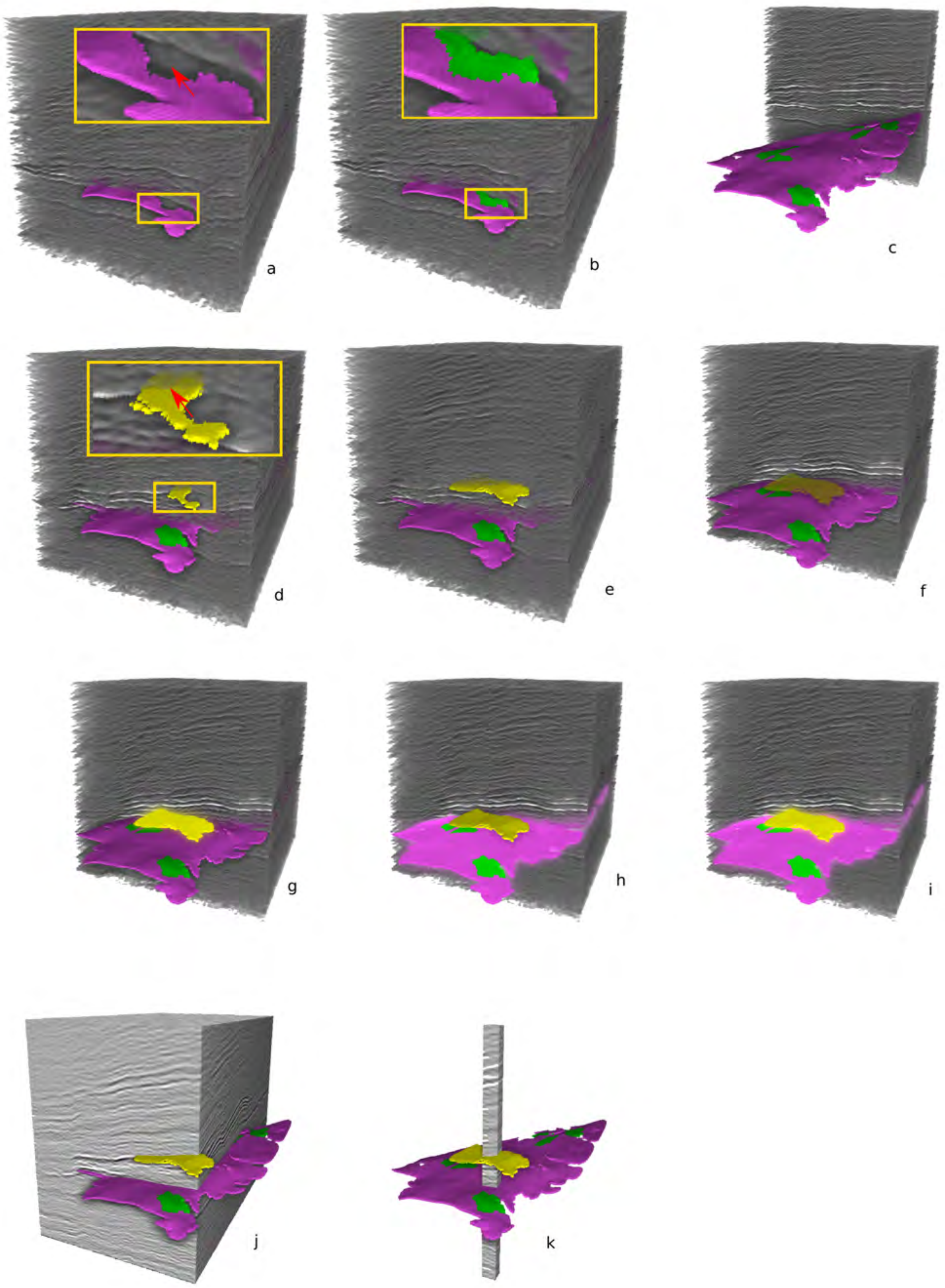


Fig. 7. The steps of interpreting two horizons, filling out holes (a-c), using different rendering styles on horizons and volume (e-i).

Chapter 9

Paper V: Knowledge Assisted Visualization of Seismic Data

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Technical Section

Knowledge-assisted visualization of seismic data

Daniel Patel [a,b,*], Øyvind Sture [a], Helwig Hauser [a], Christopher Giertsen [b], M. Eduard Gröller [c,a]

[a] *University of Bergen, Norway*
[b] *Christian Michelsen Research, Bergen, Norway*
[c] *Vienna University of Technology, Austria*

A R T I C L E I N F O

Article history:
Received 19 April 2009
Received in revised form
14 June 2009
Accepted 26 June 2009

Keywords:
Knowledge-assisted visualization
Illustrative visualization
Seismic interpretation
Rapid interpretation
2D textures
3D textures

A B S T R A C T

We present novel techniques for knowledge-assisted annotation and computer-assisted interpretation of seismic data for oil and gas exploration. We describe the existing procedure for oil and gas search which consists of manually extracting information from seismic data and then aggregating it into knowledge in a detail-oriented bottom-up approach. We then point out the weaknesses of this approach and propose how to improve on it by introducing a holistic computer-assisted top-down approach intended as a preparation step enabling a quicker, more focused and accurate bottom-up interpretation. The top-down approach also enables early representations of hypotheses and knowledge using domain-specific textures for annotating the data. Finally we discuss how these annotations can be extended to 3D for volumetric annotations.

1. Introduction

Whether we like it or not the world is dependent on energy. Oil and gas accounts for around 64% of the total world energy consumption (Iske and Randen [11]). Thus searching for and recovering these resources is important in today's society. In this paper we describe how oil and gas search is performed and we propose using knowledge-assisted visualization for improving it. There are several aspects of how the search, using seismic interpretation, is performed that makes it fit very naturally into the paradigm of knowledge-assisted visualization. A visual symbolic language for capturing knowledge has already been developed in the geosciences for interpretation. There is a high need for expressive visualizations due to large degrees of collaborative work during interpretation. Finally, large amounts of money can be saved by increasing accuracy and reducing interpretation time. We will describe in detail how these aspects of current interpretation enable knowledge-assisted visualization to accelerate the search of oil and gas.

2. Related work

The field of knowledge-assisted visualization has been described in the paper by Chen et al. [4]. We are using the meaning of data, information, knowledge and the process of knowledge-assisted visualization as defined in their paper. The entities of data, information and knowledge represent, in the order they are listed, increasingly higher degrees of abstraction and understanding. A system supporting knowledge-assisted visualization contains mechanisms for externalizing the user's knowledge gained using the system and mechanisms for explicitly expressing the knowledge in a visualization. Such a system may also contain domain-specific reasoning algorithms that assists the user in gaining knowledge. Illustrative rendering is a nonphotorealistic visualization technique using the advantages of conveying information through illustrations. This technique is well suited for representing gained knowledge and thus fits into the knowledge-assisted visualization paradigm. A good introduction to illustrative visualization and how it fits into the workflow of gaining knowledge can be found in Rautek et al. [22] and in the tutorial by Viola et al. [25]. In Rautek et al. [22] they argue for using illustrative techniques earlier in the knowledge acquisition process instead of only at the end for presenting the results as it classically has been used for.

In our work we extensively use illustrative textures to convey knowledge. Many techniques on texturing have been published. Owada et al. [17] presented an interactive system for texturing arbitrary cuts through polygonal objects. The user defines the texture flow by specifying a direction field and a distance field on the cut. The 2D texture is generated from a 2D exemplar through texture synthesis. Their method is general and therefore requires user interaction to specify the texturing. In our work we calculate a parameterization up front so texturing can be achieved quickly

* Corresponding author at: University of Bergen, PB 7803, 5020 Bergen, Norway.
E-mail address: daniel@cmr.no (D. Patel).

0097-8493/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2009.06.005

and without the need for texture synthesis. Since we target a domain-specific problem, many of the parameters defining the visualization are known prior to rendering, and less user specification is required. Lu and Ebert [15,16] generated illustrative medical renderings by applying 2D textures sampled from illustrations on volumetric data. 3D textures are created by combining color information from the illustrations with 3D clinical volume data. The synthesized 3D textures are made tileable using Wang Cubes. They do not deal with deforming the textures to follow the underlying flow of the data. Dong and Clapworthy [7] presented a technique that achieves 3D texture synthesis following the texture orientation of 3D muscle data. Their algorithm has two steps. First they determine the texture orientation by looking at the gradient data of the volume combined with a direction limited Hough transform. Second they perform a volumetric texture synthesis based on the orientation data. We identify the texture flow by using domain-specific methods as will be discussed later in this paper. The texture synthesis of Dong and Clapworthy has the drawback of not working on textures with large interior variation as textures in geologic illustrations commonly have. Wang and Mueller [26] used 3D texture synthesis to achieve sub-resolution zooming into volumetric data. With 2D images of several zoom levels of a tissue, they synthesized 3D volume textures for each level and used constrained texture synthesis during zooming to blend smoothly between the levels. They address the issue of sub-resolution details but do not consider texture flow. Kopf et al. [13] created convincing 3D textures from 2D texture exemplars where they are also able to synthesize semitransparent 3D textures. The 3D textures are created by gradually optimizing the similarity in local neighborhoods between the synthesized solid texture and the exemplar. Global similarity is achieved by color histogram matching from the 2D exemplar. For synthesis control different 2D exemplars may be specified for each of the orthogonal views of the wished solid texture. Due to the stochastic nature of the synthesis, full control is not possible and unexpected 3D textures may be generated. Stochastically synthesized textures may have low similarity to the original exemplar and are not ideal for our use as texture recognizability is important in our approach.

Much work has been done in multiattribute visualization. Bürger et al. [2] presented a state of the art overview. Crawfis and Allison [6] presented a general framework where textures, bump maps and contour lines are used for multiattribute visualization. Kirby et al. [12] presented multiattribute visualization of 2D flows using concepts from painting. Taylor [24] took a general approach by combining color, transparency, contour lines, textures and spot noise. He succeeded in visualizing four attributes simultaneously. However, little work has been done in multiattribute visualization of seismic data as we focus on in our works.

Ropinski et al. [23] covered volume rendering of seismic data in VR. They presented spherical and cubic cutouts which have a different transfer function than the surrounding volume. We incorporate and extend this concept in our work for combining illustrative visualization with scientific volume visualization inside cutouts. Plate et al. [21] and Castanie et al. [3] discussed the handling of large seismic volumes. Commercial software used in oil companies includes HydroVR [14] and Petrel [1]. None of these works discuss illustrative techniques or top-down interpretation as presented here. Horizon interpretation algorithms are typically semiautomatic requiring detailed and time consuming user involvement. Pepper and Bejarano [20] gave an overview of computer-assisted interpretation algorithms for seismic data. Several growing algorithms exist. Castanie et al. [3] proposed user seeding followed by growing based on the local waveform of the seedpoint. Interpretation software [1] performs growing in voxels that have been thresholded or in extrema or zero crossings of the local waveform. For horizon extraction algorithms the user typically sets growing parameters and seeds and grows until a satisfactory result is obtained. The parameters to be set are related to the internal growing/image processing algorithms and can be difficult to understand. One might argue that the user is given too much low-level control. In our system we aim at avoiding the need for parameter tweaking in early exploratory phases by offering automatically pre-grown horizon candidates. A similar idea was developed by Faraklioti and Petrou [9] but which results in fewer detected and more fragmented horizons due to the requirement of horizon planarity.

3. Overview

In Section 4 we first describe the current method of oil and gas search. We then point out the weaknesses of this approach and propose how to improve on it by introducing a holistic and sketch based top-down approach. The top-down approach is intended as a preparation step enabling a quicker, more focused and accurate bottom-up interpretation. It enables early representations of hypotheses and knowledge by providing domain-specific annotations of the data using textures. The section concludes with a comparison of the old and the proposed new approach. In Section 5 we provide examples of annotated seismic data used in the top-down approach. In Section 6 we present a taxonomy of texturing techniques that can be used for annotating seismic volumetric data and their associated advantages and disadvantages. The article is rounded up with conclusions in Section 7.

4. Bottom-up and top-down interpretation

In this section we describe the current bottom-up interpretation pipeline, introduce our new top-down methodology and explain some of the automated interpretation techniques enabling the top-down approach. Afterwards we compare the two approaches.

4.1. Bottom-up interpretation

Seismic volumetric reflection data are used when exploring a geological prospect. The data are gathered by sending sound waves into the ground and processing the echoes. In the collected seismic data, hydrocarbon traps are searched for by looking for certain structural signatures. The first step in the interpretation is to get a structural overview of the data by identifying horizons, which are separations between rock layers, and faults, which are cracks and discontinuities in the rock layers.

Currently for volumetric data, interpretation is performed by a single person, slice by slice. The domain expert is looking at the data at maximum resolution and in detail tracing out structures manually. For more details about the interpretation pipeline, see Iske and Randen [11] and Patel et al. [18]. We have exemplified the interpretation process in Fig. 1. At the top of Fig. 1a, one can see a specific seismic slice under interpretation. The interpreter traces out horizons $h_1 - h_6$ and faults f_1, f_2. Then as more insight is acquired, the interpreter tries to connect the identified horizon pieces across faults (f_1, f_2) and through noisy areas (stippled lines between h_5 and h_6). Pieces he thinks belong together are shown with the same color. The conceptual pyramid below the slice illustrates this aggregation of insight. At the lowest level in the tree inside the pyramid, horizons and faults are identified with no relation to each other. At consecutive higher levels, the horizons are grouped. The groupings are represented with common nodes, and their interaction with faults is identified. Horizon and fault interactions are depicted as curved edges from fault nodes to horizon group-nodes. When moving upwards in the pyramid, several lower level structures are grouped into fewer higher level structures. The result is that the amount of data is reduced, which is indicated by the narrowing pyramid shape, while insight or knowledge increases. When an overview has been reached, represented by the single node at the top of the pyramid, an expert meeting is arranged. A correct interpretation requires expertise in several fields. Therefore the overview is discussed by an interdisciplinary team of geologists, geophycisists, well engineers and other domain experts. When high-level structures can be seen in relation to each other, errors in the interpretation are more easily identified. The discussions will in many cases conclude that parts of the structures or interrelations between structures are incorrectly classified. As a consequence the prospect must now undergo a time consuming reinterpretation (Fig. 1b). A problematic region is indicated with an ellipse in the slice of Fig. 1a. The interpreter incorrectly connected horizons h_1 and h_3 across fault f_1 while they should be considered as separate horizons. Interpretation errors in lower levels can easily propagate to higher levels. Therefore a reinterpretation is made after ungrouping horizons h_1 and h_3. A different color is assigned to the group of horizons connected to h_3 in Fig. 1b due to the ungrouping. When the multidisciplinary team agrees on the reinterpretation, a more accurate understanding of the seismic data has been gained and areas of particular interest with regard to hydrocarbon potential are identified. They are indicated with ellipses in the slices of Fig. 1b and c. The interpretation is then focused on this area and is finally sent to a seismic illustrator. The illustrator manually generates illustrations that capture the gained high-level knowledge in a standardized and easy to understand way. Making such illustrations is quite a time consuming task. The process just described is what we refer to as the bottom-up approach.

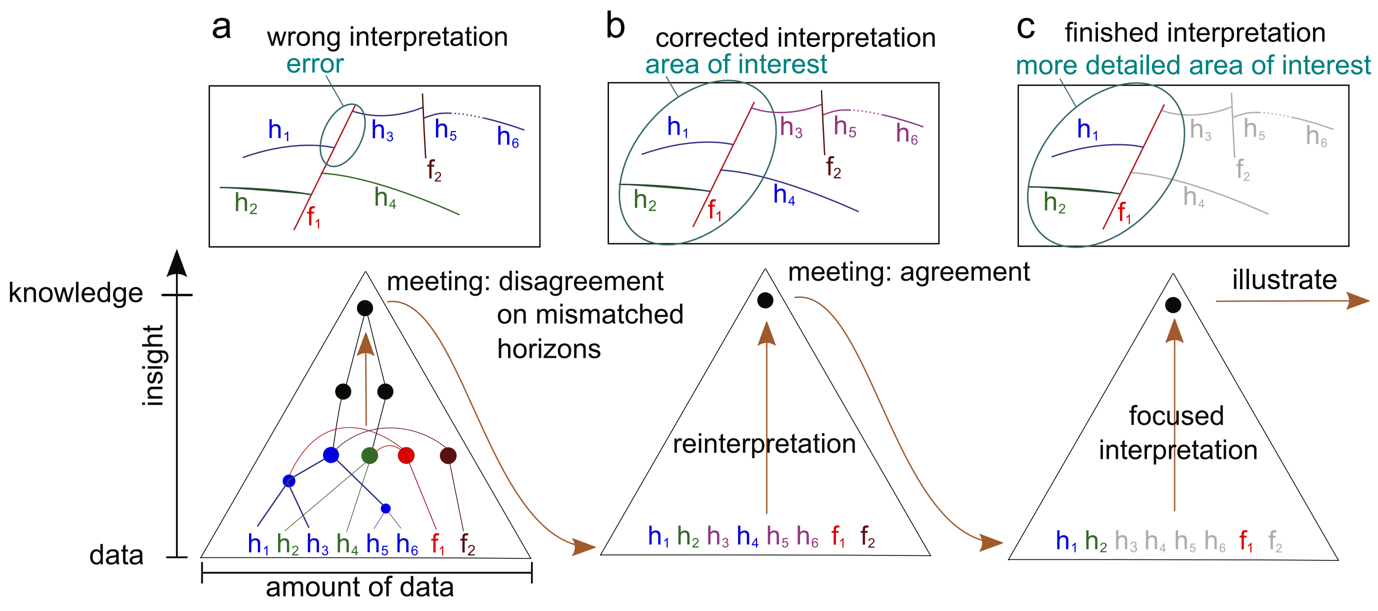


Fig. 1. An example of the steps performed during a bottom-up interpretation.

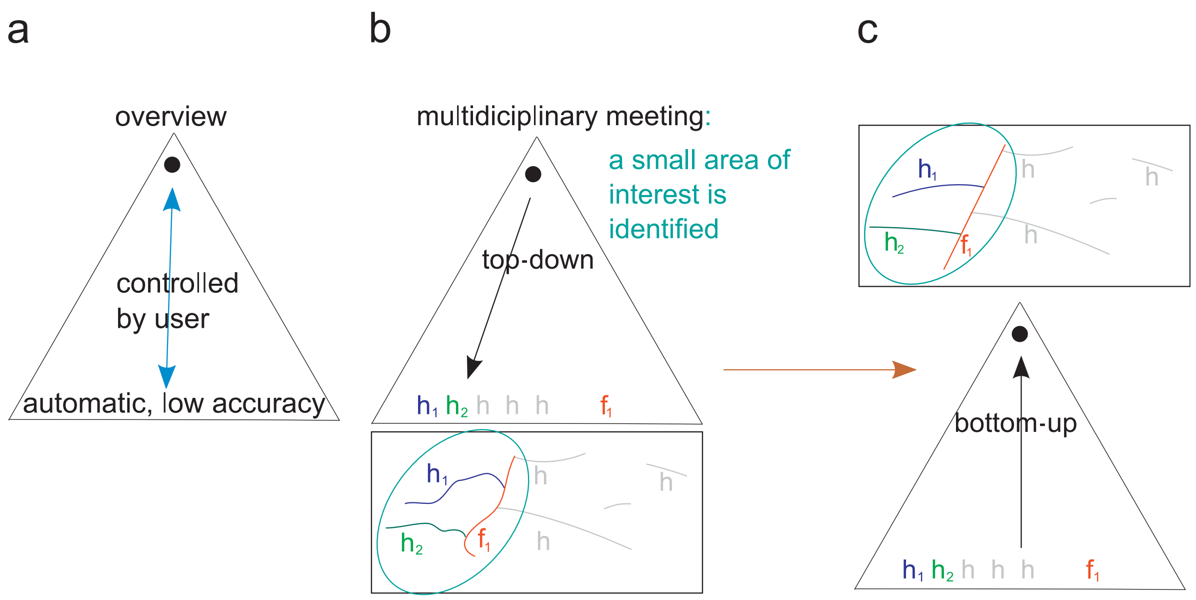


Fig. 2. The steps performed during a top-down followed by a bottom-up interpretation. An overview of the conceptual pyramid is given in (a). In (b) a top-down interpretation is performed prior to the bottom-up interpretation in (c).

The bottom-up approach does not scale well with the amount of data. With the rise of data sizes and resolutions due to technological advances, this approach is increasingly time consuming as it works on the highest level of resolution. Seismic volume data are inaccurate due to the complex and underdetermined process of transforming the reflected sound waves into a 3D volume. Further inaccuracies stem from physical phenomena such as noise, multiple reflections and acoustic shadowing. Our top-down approach aims at improving the weak points of the bottom-up approach. Firstly, it is problematic to start with an *accurate and detailed* interpretation of uninterpreted seismic data due to their *inaccuracies and uncertainties*. Secondly, expert guidance from the multidisciplinary team comes late, after the time consuming detailed interpretation for the overview has been made. Due to the too late guidance, reinterpretations must often be performed. Ultimately, we wish to reduce the time necessary to create the final illustration and to allow for illustrations to be created at any stage of the interpretation.

4.2. *Top-down interpretation*

We propose to address the above mentioned problems by introducing a quick top-down interpretation stage at an early point in time. This is performed collaboratively by a multidisciplinary team of interpreters before the slower single person bottom-up analysis takes place (see Fig. 2). In the top-down stage, interpretation begins at a high level in the conceptual pyramid on a coarse level of detail by looking at the data with a highly zoomed out and abstracted view. The approach uses computer-assisted sketching and multiattribute visualizations for expressing and discussing hypotheses at an early stage during interpretation. To present hypotheses and knowledge, illustrative rendering is used. Illustrative techniques have been developed for the purpose of communicating high level aspects in a simple way. Elements of interest are illustratively emphasized to represent the knowledge acquired during the interpretation process. Illustrative methods allow for annotating the data so that the resulting image closely communicates the interpreters' internal models and hypotheses. This enables the interpretation team to clearly communicate and get a common understanding of each others ideas. When the understanding at the current level of detail is agreed on, a more detailed level can be investigated. A more detailed level is gained by either zooming in on a specific area of the data or by adding more data to the visualization by including new attributes. Since illustrative visualizations are created at all interpretation stages external communication outside the team is possible at any time. The bottom-up approach starts only after an agreed rough overview of the data has been made as seen in the pyramid of Fig. 2b. The gain of adding the top-down stage is to focus the bottom-up interpretation on important structures. This avoids interpreting areas of low or no interest, such as the unnecessarily interpreted gray areas in Fig. 1c. Furthermore uncertainty and the need for reinterpretation at late stages due to disagreements in the team is reduced. The top-down stage can also act as a screening to find out early if the prospect lacks potential hydrocarbon structures and should be abandoned.

In Fig. 2 the workflow of performing a top-down interpretation prior to the bottom-up interpretation is shown. We indicate the inaccuracies and sketchy nature of the top-down interpretation by using wiggly horizons and faults in the pyramid of Fig. 2b. These are concretized in pyramid Fig. 2c after a bottom-up interpretation confined to the identified area of interest was performed. Outside the area of interest, an even more sketchy and inaccurate interpretation was performed for overview reasons. Although the process is top-down from the user's perspective, it is bottom-up from a computational perspective since it builds on the automatically extracted low-level structures at the bottom of the pyramid.

There are two mechanisms that enable top-down interpretation. The first is to relieve the user from performing time consuming low-level interpretation of structures. This is achieved by automatically preprocessing the data for identifying low-level structures. This mechanism is represented as the text at the bottom of the pyramid in Fig. 2a. The process can be seen as a computer-driven interpretation. However a computer-driven interpretation may generate inaccurate structures and interpretation suggestions as it cannot match the accuracy and experience of a human interpreter. The user must therefore be given the opportunity to quickly and easily browse through the computer-generated suggestions to select valid ones and aggregate them into higher level structures. This is the second mechanism, which is indicated by the blue arrow in the pyramid of Fig. 2a. The first mechanism is implemented by automatically deriving appropriate support data in a preprocessing stage before interpretation. The second mechanism is implemented by giving the user the control to browse the computer-generated support information and quickly create abstracted representations of the data. As opposed to raw data visualization, abstracted representations can be sparse, i.e. not cluttering and covering up the view. This yields two ways of aggregating data and getting overviews—by zoom-outs and by multiattribute renderings. Fig. 3 has examples of zoom-outs in the middle column and an example of multiattribute rendering is shown in the right image.

4.3. *Automatic interpretation*

Horizon candidates are automatically extracted in a preprocessing stage by adapting the method described in Iske and Randen [11]. Horizon candidates are identified by tracing out line segments following the valleys and ridges of the height field defined by the reflection values on a slice, see Fig. 4. The result is a collection of lines going through the horizons of the slice as seen in Fig. 4c. Our method differs from existing horizon tracing algorithms found in commercial software as it does not require a user defined seed point for each trace. This avoids having the interpretation flow interrupted due to setting seed parameters and waiting for growing results.

For each traced horizon line, we calculate measures like length, average strength and average angle. The user can filter horizons based on any of these measures. It is also possible for the user to select a subset of the horizons. Picked or filtered horizons can be visualized as line segments or texturing can be applied for annotation reasons. Each horizon line has defined a segmentation mask around it where texturing or coloring can take place. Transparency can be set so that the original seismic data will be visible underneath. The texture mapping is defined by a parameterization created from the extracted horizon lines. This is done to ensure consistent deformation and orientation of the textures with the underlying seismic data. For details about the parameterization, see Patel et al. [18].

4.4. *Comparison*

In Fig. 5a we compare the bottom-up and the top-down approach. The vertical axis represents the information level of the structures that have been identified in the interpretation. The black dot close to the origin represents the lowest level of information, being the raw seismic reflection-values. The black dot above represents the highest degree of information, where horizons, faults and other structures have been interpreted and

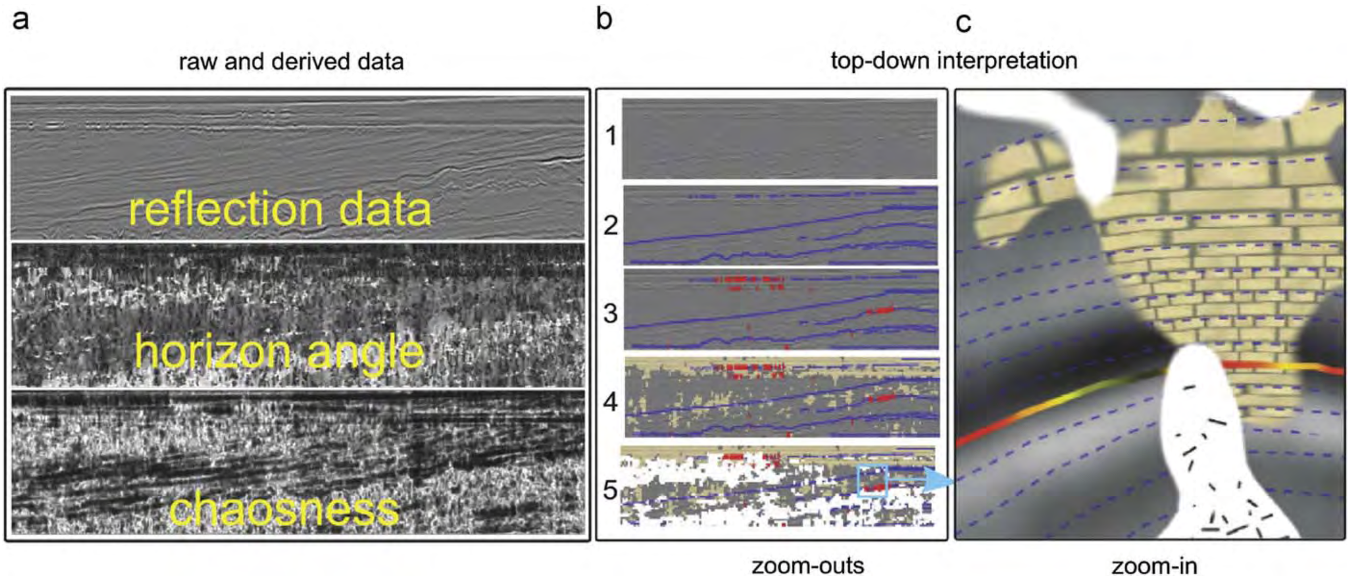


Fig. 3. Frame (a) shows the input data to be interpreted. Images 1–5 in frame (b) show the consecutive adding of illustrative layers on a highly zoomed-out slice as part of a top-down interpretation. Image 1 shows a zoom-out of the reflection data. Image 2 shows automatically generated lines that reveal the trends of the horizons. In image 3 areas of strong reflection values are shown with red lines. In image 4 areas of low-angled horizons are shown in brown corresponding to dark areas in the 'horizon angle' slice. Image 5 shows areas of badly defined horizons in white corresponding to high values in the 'chaosness' slice. Frame (c) shows a zoom-in on the indicated region of image 5.

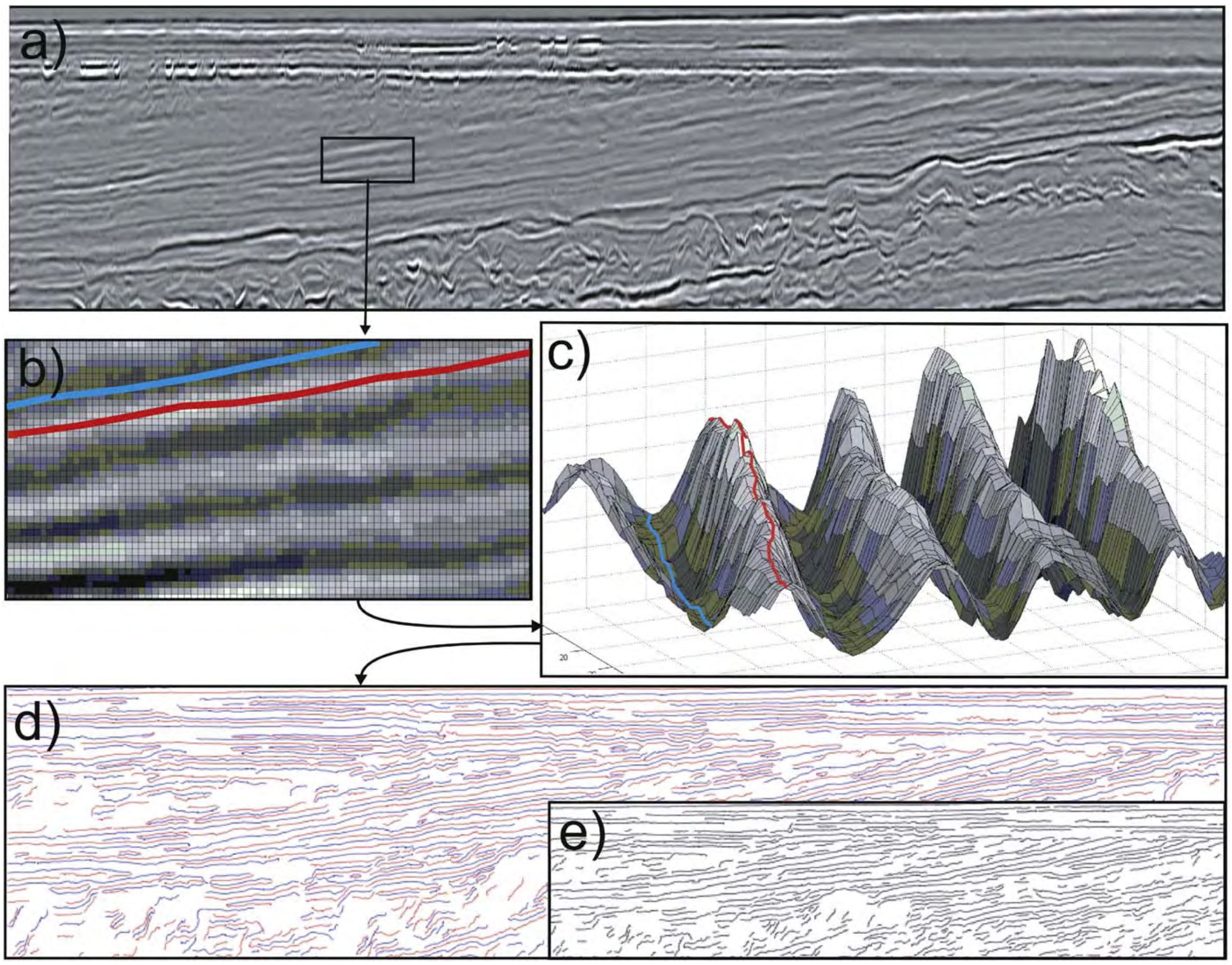


Fig. 4. Extracting horizons by tracing along peaks and valleys in the reflection data in (a). A peak is marked with red and a valley is marked in blue in (b) and (c). The rectangle in image (d) shows all extracted peaks and valleys from (a). Image (e) is a zoom-out of (d) with fewer horizons displayed to avoid crowdiness to demonstrate how an abstracted data representation enables sparse information visualization.

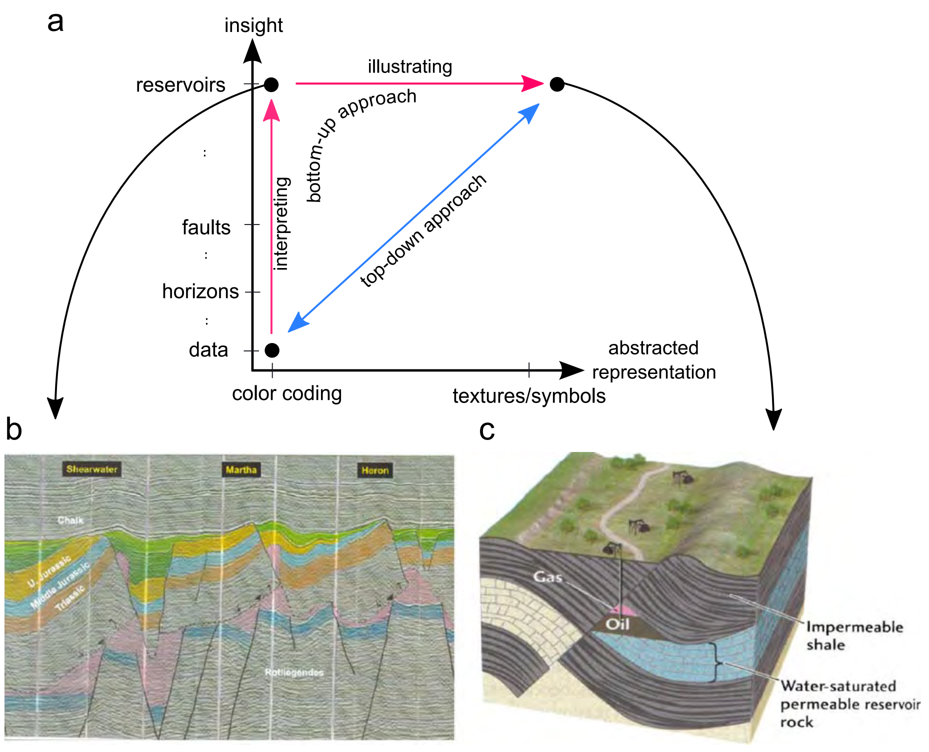


Fig. 5. Comparison of the bottom-up interpretation with our top-down approach. Image (b) is from Emery and Myers [8]. Image (c) is from Grotzinger et al. [10].

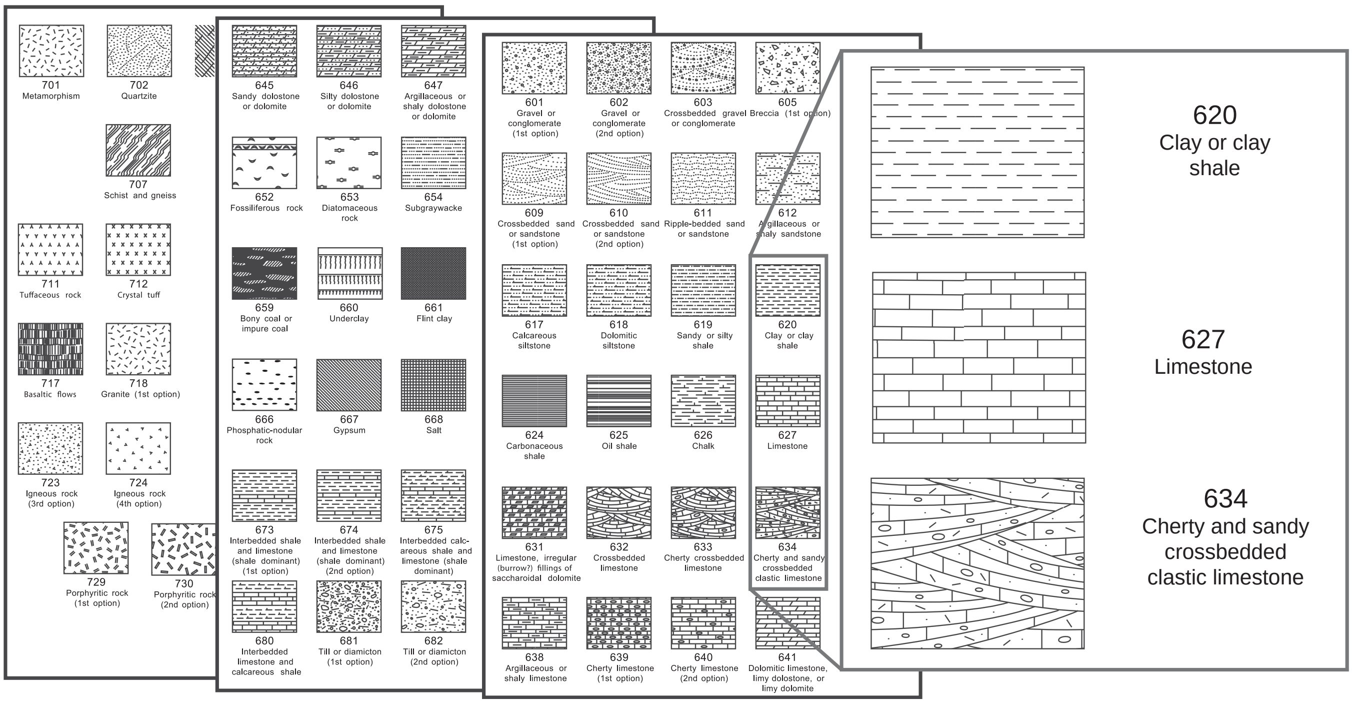

701 Metamorphism
702 Quartzite
707 Schist and gneiss
711 Tuffaceous rock
712 Crystal tuff
717 Basaltic flows
718 Granite (1st option)
723 Igneous rock (3rd option)
724 Igneous rock (4th option)
729 Porphyritic rock (1st option)
730 Porphyritic rock (2nd option)
645 Sandy dolostone or dolomite
646 Silty dolostone or dolomite
647 Argillaceous or shaly dolostone or dolomite
652 Fossiliferous rock
653 Diatomaceous rock
654 Subgraywacke
659 Bony coal or impure coal
660 Underclay
661 Flint clay
666 Phosphatic-nodular rock
667 Gypsum
668 Salt
673 Interbedded shale and limestone (shale dominant) (1st option)
674 Interbedded shale and limestone (shale dominant) (2nd option)
675 Interbedded calcareous shale and limestone (shale dominant)
680 Interbedded limestone and calcareous shale
681 Till or diamicton (1st option)
682 Till or diamicton (2nd option)
601 Gravel or conglomerate (1st option)
602 Gravel or conglomerate (2nd option)
603 Crossbedded gravel or conglomerate
605 Breccia (1st option)
609 Crossbedded sand or sandstone (1st option)
610 Crossbedded sand or sandstone (2nd option)
611 Ripple-bedded sand or sandstone
612 Argillaceous or shaly sandstone
617 Calcareous siltstone
618 Dolomitic siltstone
619 Sandy or silty shale
620 Clay or clay shale
624 Carbonaceous shale
625 Oil shale
626 Chalk
627 Limestone
631 Limestone, irregular (burrow?) fillings of saccharoidal dolomite
632 Crossbedded limestone
633 Cherty crossbedded limestone
634 Cherty and sandy crossbedded clastic limestone
638 Argillaceous or shaly limestone
639 Cherty limestone (1st option)
640 Cherty limestone (2nd option)
641 Dolomitic limestone, limy dolostone, or limy dolomite
620 Clay or clay shale
627 Limestone
634 Cherty and sandy crossbedded clastic limestone

Fig. 6. Examples from a geological texture library [5] for annotating knowledge. Three of the textures are enlarged in the far right rectangle.

potential oil reservoirs have been found. An example of such an interpretation is seen in Fig. 5b. The horizontal axis in Fig. 5a indicates how abstracted information is represented. The further right, the more communicative the representation is, with textures and symbols vs color-coded data represented by the black dot close to the origin. An example of such a representation is given in Fig. 5c. A bottom-up interpretation is given by the vertical red line. It makes little or no use of illustrative techniques during interpretation. This is indicated by having the red vertical line far to the left. After the interpretation, communication-friendly illustrations are generated. This is indicated by the red horizontal line at the top. Both interpreting and illustrating requires considerable manual work. In contrast, during a top-down interpretation, structures are found with computer assistance and the structures can be shown with automatic illustrative techniques. Computer-assisted interpretation enables vertical movements in the space of Fig. 5a and illustrative techniques enable horizontal movements. Creating a communication-friendly interpretation is faster and has the advantage that the communicative content grows along with the increased knowledge. Thus the interpreter and the team can take advantage of the illustratively abstracted and communicative representation during analysis. This is in contrast to only having the representation available after both the interpretation is finished and the manual illustration has been made.

5. Representing knowledge with 2D textures

To ease communication, geologists and geoillustrators use a standardized language for representing knowledge. This language consists of textures for representing rock types and other information. The US Federal Geographic Data Committee (FGDC) has produced a document [5] with more than 100 standardized textures for rocks. These textures are referred to as lithological symbols in the geosciences. Fig. 6 shows three pages from this geological texture library. The oil company we have been collaborating with uses a colored version of the FGDC standard with some minor variations as shown in Fig. 7. Similarities of the two standards can be seen in the clay shale and the limestone textures in Figs. 6 and 7. Textures have the advantage that they can give an integrated visualization of layers, faults and attributes. The layer type and its orientation is communicated locally at any point on the texture. The way sediments have deposited and created the subsurface layers is very important to model during interpretation. For expressing such models, oriented textures are central. Finally, textures do not show sampling artifacts but are aesthetically pleasing even after extreme zooming, and visualizations can be more understandable by non-experts.

5.1. Computer-assisted annotation of knowledge

We create illustrative and knowledge-representing renderings by defining separate layers for each aspect of knowledge that is to be communicated. The illustrative layers are then composited into one illustration. Each illustrative layer is created by a user defined mapping from raw data or processed data as seen in Fig. 3a to an abstracted representation. By mapping the data to the standardized representations used in the geosciences, the user creates a rendering that encodes seismic domain knowledge. The representations are either textures or lines that follow the trends of the underlying seismic data. Value ranges of the seismic data are mapped to different types of textures. Line styles have a user defined sparseness and transparency. See Patel et al. [18] for details on how the illustrative layers are specified.

To achieve the effect of textures and lines following the orientation trend of the underlying reflection data, we create a specific parameterization from the traced horizons. The parameterization ensures that the illustrative textures and straight lines are aligned with the extracted horizons. Illustrative layers in combination with sparseness control are key in performing a top-down interpretation. By using the appropriate sparseness and the appropriate number of layers, communicative illustrations can be made for any zoom level. An example of an overview picture using these techniques can be seen in Fig. 3. The raw data, i.e. reflection data, is given in the top slice of Fig. 3a. The two slices below are derived from the top slice. Other papers [11,18] give more information about derived attributes. A bottom-up interpretation would require that the user works on highly zoomed-in views of these slices and switches back and forth between them. Trying to get an overview of the reflection data by zooming out results in image 1 in Fig. 3b where few details are visible. However, by adding illustrative layers (images 2–5) with a sparse drawing style defined by the different modalities, it is possible to get a multiattribute overview of the data on a zoomed-out image. After an overview has been achieved, the interpreter can look closer at an area of interest (Fig. 3c).

In Fig. 7 another example of computer-assisted knowledge-annotation of seismic data is shown. The seismic data have been divided into rock layers and assigned textures representing rock

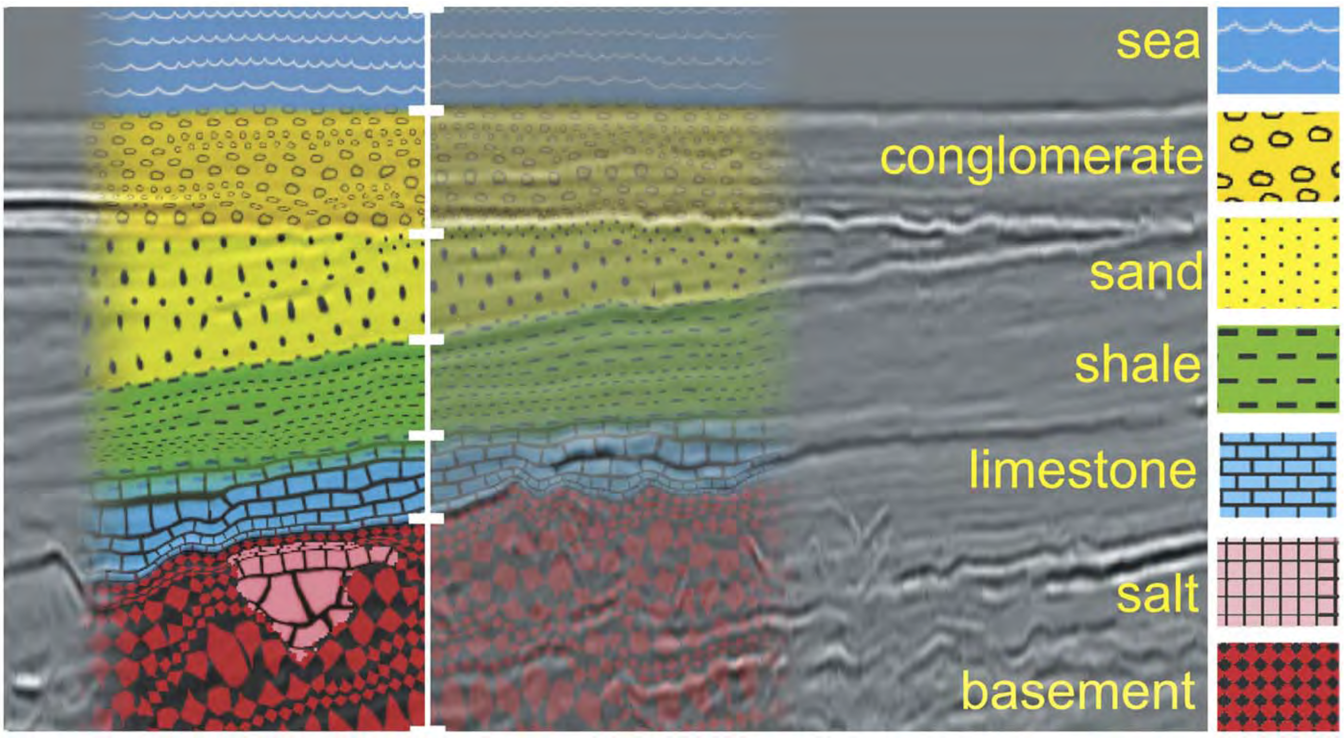


Fig. 7. A slice annotated by an interpreter. The legend to the right and the vertical white line in the middle has been laid over our original rendering. Low transparency is shown on the left side of the white line and high transparency is shown on the right side for demonstration purposes.

types. With computer-assisted annotation the illustrations can be created very fast. The interpreter only needs to assign different textures to depths along a vertical line through the slice (white line). Afterwards the textures are dragged out horizontally, to a user defined width, following the horizon trends. This is enabled by exploiting the calculated parameterization originally used for texturing. The salt area at the bottom is annotated by selecting precalculated horizon patches through mouse picking and assigning a specific texture to the horizon selection. The transparency of the overlaid texturing can be changed so the underlying seismic can be seen in combination with the proposed rock subdivision for verification reasons. Two different transparencies are shown on the left and right side of the white line in Fig. 7. For more details, see Patel et al. [18].

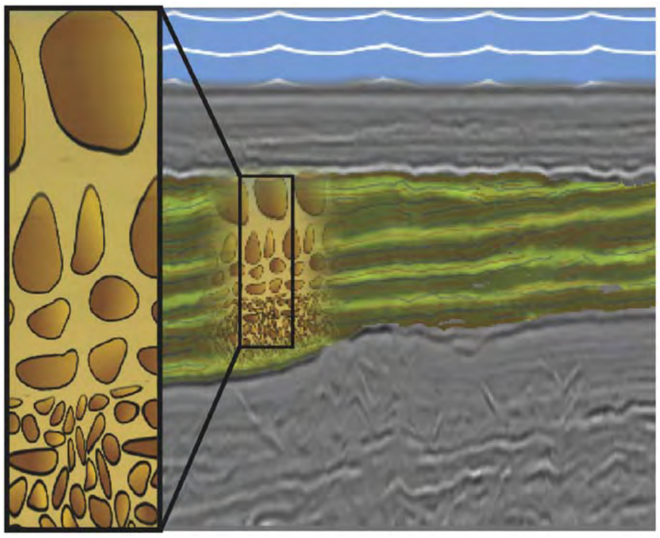


Fig. 8. Annotated seismic slice. Zoom-in on a texture showing sand grains with decreasing size along depth which represents a specific hypothesis.

A different example of computer-assisted annotation is shown in Fig. 8. The green area was annotated through mouse picking in the same way as the salt area in Fig. 7. The precalculated and selected horizon patches result in a green texture. The horizons themselves can be seen as faint red and blue lines corresponding to the peaks and valleys described in Fig. 4. The green textured area corresponds to the sand area in Fig. 7. The interpreter believed that this geologic layer was created in a sedimentation process and consequently formed a hypothesis that the sizes of the sand grains decrease with depth. To annotate this, different sand textures were assigned along the depth of the geologic layer and the textures were slightly dragged out horizontally. The sand textures are enlarged in the close-up view of Fig. 8.

With our techniques it is possible to smoothly move back and forth between visualization of data and visualization of knowledge as demonstrated in Fig. 9. The bottom image is a color-coded rendering of an impedance volume which has been calculated from the reflection volume. The impedance reveals the speed of sound through the rock. Blue denotes low impedance and green denotes high impedance. The top image shows the interpreted data with different textures for each rock layer. The middle image is a blend of the top and the bottom image where impedance can be seen in relation to the interpreted layers. On the right side of the middle image a correspondence between high impedance (green color) and its confinement inside one layer can be seen. Blendings make intermediate representations possible where both modes are displayed simultaneously. It enables the

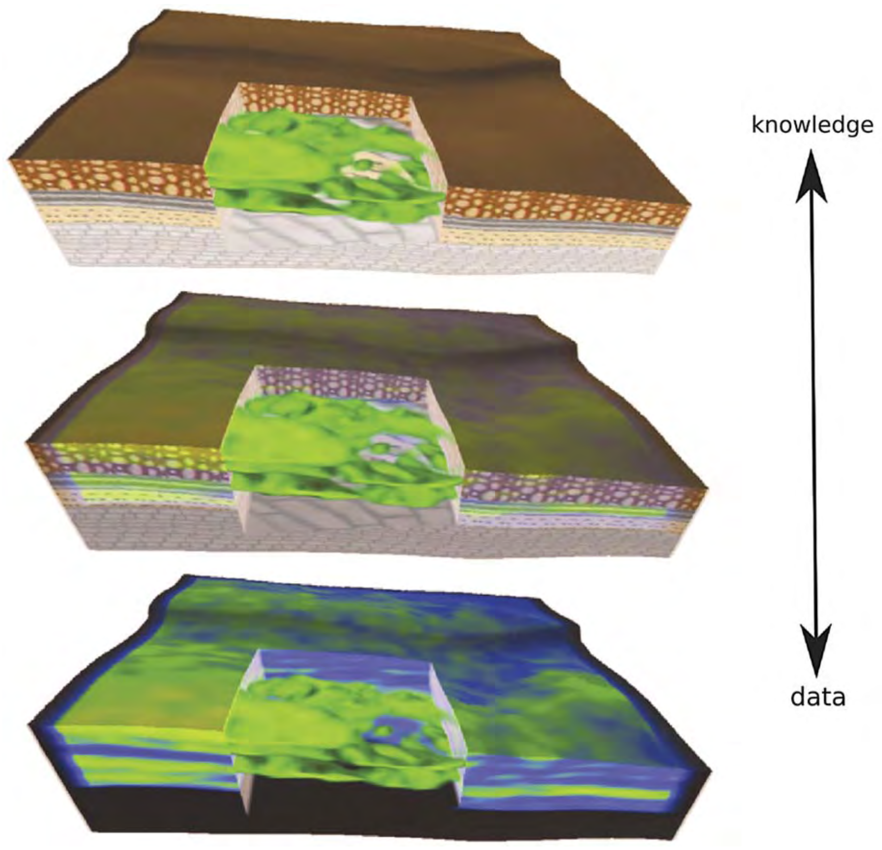


Fig. 9. The image series shows the transition from raw data visualization at the bottom to abstracted illustrative visualization at the top [19].

user to compare and verify the interpretation with the original data. More details can be found in the work of Patel et al. [19].

6. Representing knowledge with 3D textures

All standardized geologic textures are defined in 2D. However the Earth crust that is to be interpreted is inherently 3D. 2D textures do lend themselves directly to illustrate 2D seismic slices but have limitations when applied on 3D data as will be discussed in this section. On 3D data, 2D textures are frequently applied on planar cross sections. This technique is used in 3D geological illustrations. Several examples are given in Grotzinger et al. [10]. In this section we investigate using 3D textures for annotating 3D seismic data. Several advantages can be gained if suitable 3D textural representation of the 2D lithological symbols are specified and used. From an algorithmic point of view it is simpler to map 3D textures to 3D volumes and surfaces than 2D textures. From a perceptual point of view, 3D textures will reduce the problems of spatial and frame-to-frame incoherencies as will be discussed in the next paragraph. Additionally 3D semitransparent textures may give rise to a higher perceptual depth. Volumetric variations can be revealed within the 3D texture and not only on the exterior boundary as is the case when using a 2D texture. Therefore in the context of knowledge-assisted visualization of 3D seismic data and as an outlook in the future, we explore 3D seismic textures.

6.1. 3D texture examples and comparison with 2D

Using 2D textures on 3D data as presented in this paper has several limitations. The dimensional mismatch between 2D textures and the 3D volume to apply textures on can lead to spatial incoherencies. 2D seismic textures are typically mapped to planar surfaces in 3D. Distortion problems arise with mappings to curved surfaces, and frame-to-frame incoherencies arise when interactively moving a textured cut-plane. In addition, using 2D textures on semitransparent volumetric regions is not well defined due to the dimensional mismatch. These problems can be solved by using appropriate 3D textures instead. However the

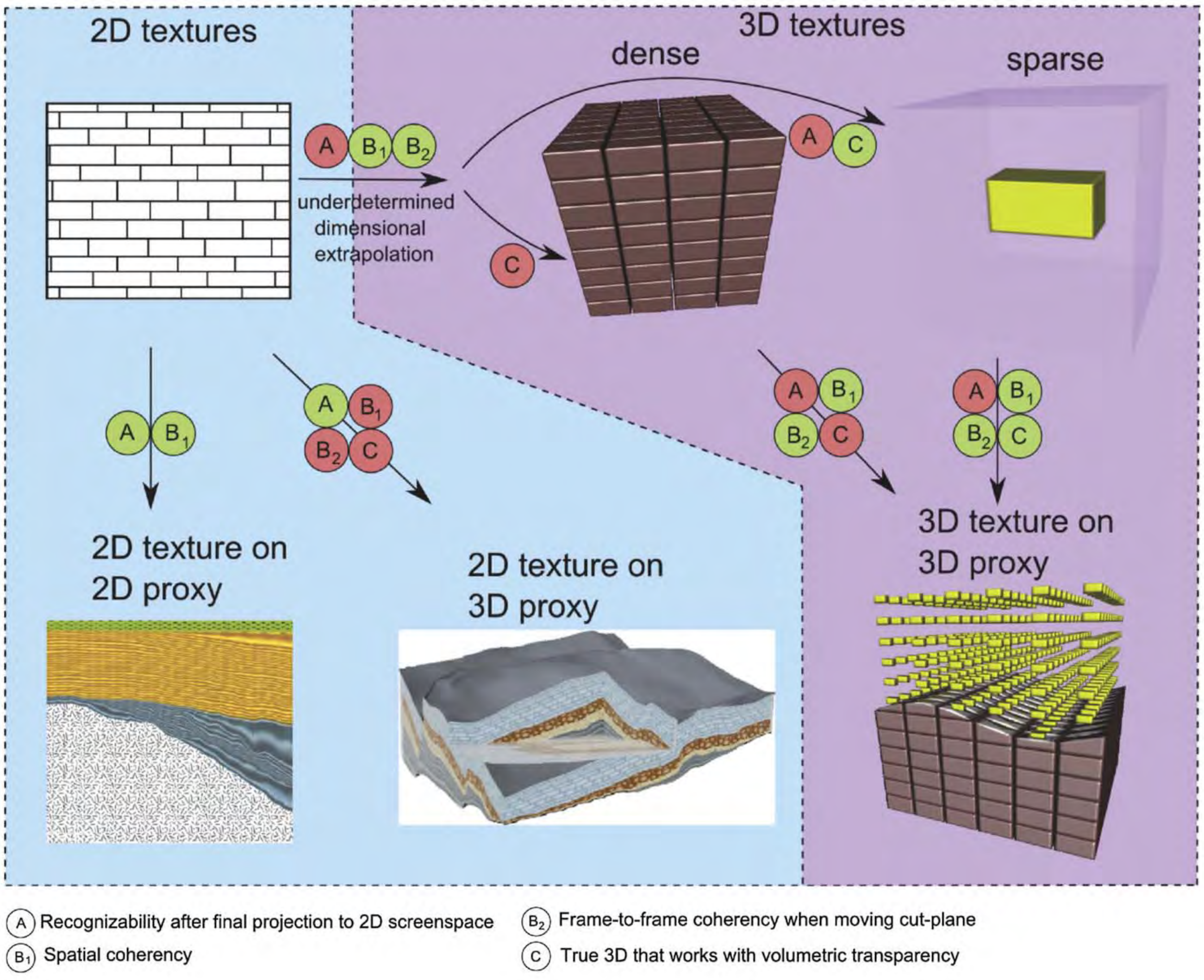

2D textures
3D textures
dense
sparse
A B1 B2
underdetermined dimensional extrapolation
C
A C
A B1
A B1 B2 C
A B1 B2 C
A B1 B2 C
2D texture on 2D proxy
2D texture on 3D proxy
3D texture on 3D proxy
A Recognizability after final projection to 2D screenspace
B1 Spatial coherency
B2 Frame-to-frame coherency when moving cut-plane
C True 3D that works with volumetric transparency

Fig. 10. An overview of advantages and disadvantages when using 2D vs 3D dense vs 3D sparse textures. The top row shows a 2D, a 3D dense and a 3D sparse texture. The arrows from left to right represent transformations of textures to 3D. The arrows from the top to the bottom row represent mappings of textures to geometries. The encircled characters on the arrows refer to perceptual advantages (green) and disadvantages (red) when performing these transformations and mappings. They are defined below the figure.

textures used for conveying knowledge in geosciences are only defined in 2D, there exist no 3D versions. If we want to use 3D textures, we must synthesize them ourselves from their 2D counterparts. Extending 2D textures to 3D has been attempted in other domains as discussed in the related-work section. However no work has been done on synthesizing 3D seismic textures and little work has been done on 3D textures with transparency in the context of scientific visualization. A dimensional extrapolation from a 2D texture to a 3D volumetric texture is underdetermined and has several possible mappings. We have looked at automatic algorithms for performing the extrapolation. They often have shortcomings when it comes to regular textures and in general when there is a specific mental expectation of how the 3D shape should look like that cannot be specified to the texture synthesizer. For instance creating a 3D texture from a 2D brick texture according to the algorithm of Kopf et al. [13] does not create a 3D cuboid brick texture as expected. Such algorithms take into account statistical properties and further constraints. They for example assume uniform distributions which are not always given for certain coherent structures.

Since the 2D–3D mapping is underdetermined it can result in dissimilarities between the rendering of a synthesized 3D texture and the rendering of the original 2D texture. Dissimilarities are unwanted as it is important that the viewer can relate a rendered 3D texture to its 2D origin. This is because the 2D texture has a connotation and represents knowledge. Thus there are respective advantages and disadvantages for either using 2D textures directly on surfaces in 3D or translating them to 3D and using them as volumetric textures.

We differentiate between dense 3D textures having no regions of full transparency, and sparse 3D textures having sparse opaque structures within transparent regions. We make this distinction due to their different perceptual characteristics. With dense 3D textures it is difficult to see the interior; however slices through these textures can be very similar to their 2D origin. With sparse 3D textures one can see the interior or can even see through which might reveal more information.

In Fig. 10 we list the advantages and disadvantages when applying 2D, 3D dense and 3D sparse textures on 2D and 3D regions. At the top of Fig. 10, examples of the three types of textures are given. The 3D dense and sparse textures will have other perceptual properties than the 2D exemplars they are synthesized from. Four perceptual characteristics, i.e., A, B_1, B_2 and C, are specified below the figure and listed on the transitional arrows between the texture types in the upper part of the figure. We refer to a 2D surface to be textured as a 2D proxy and to a 3D volumetric region to be textured as a 3D proxy. Textured 2D and 3D proxies are shown in the lower part of the figure. Different perceptual properties arise depending on the texture and proxy combinations. They are listed on the transitional arrows going from the top part with textures to the bottom part with proxies in Fig. 10. Gaining a property is annotated in green and losing a property is annotated in red. 2D textures applied on 2D proxies keep the textures' original appearance when disregarding rotational and perspective distortions, so the textures are recognizable (A). Problems with spatial coherency only show up on 3D textures thus property B_1 is present. The properties B_2 and C are not listed since they are undefined in 2D.

Applying 2D textures on 3D proxies in essence means applying 2D textures on 2D proxies positioned in 3D space. Therefore property A is also satisfied here. However the textures on the 2D proxies are not aware of their 3D embedding. This leads to inconsistent texturing on adjoining surfaces and on curved surfaces. Property B_1 is thus not satisfied for 2D textures on 3D proxies. The mapping of 2D textures is not synchronized with their 3D embedding. This leads to inconsistent texturing from frame-to-frame when interactively moving a 2D proxy such as a slice plane in 3D space. Thus property B_2 is not satisfied either. Finally, since the textures are not 3D they cannot be rendered in a meaningful way with volumetric transparency, therefore property C is also not satisfied.

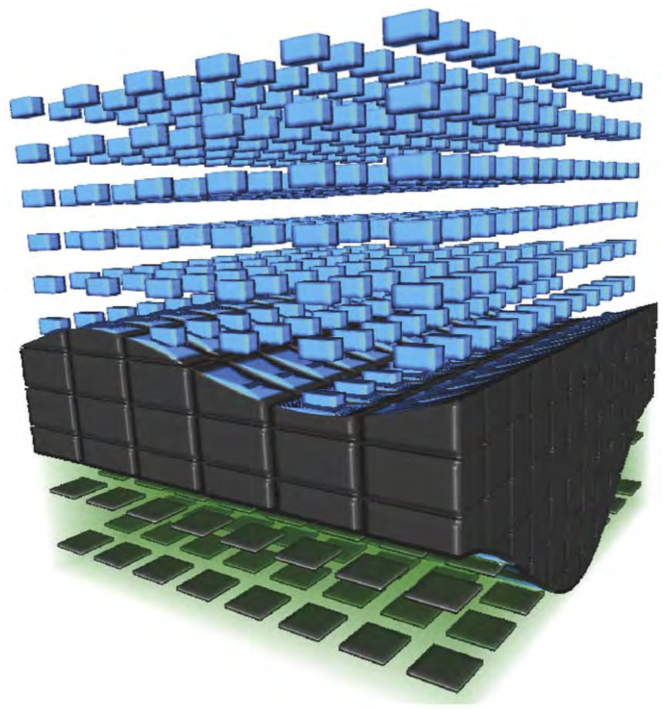


Fig. 11. An attempt at transforming the 2D limestone and shale seismic textures of Fig. 7 to 3D textures. The top layer is a sparse limestone 3D texture. The middle layer is a dense limestone 3D texture and the bottom texture is a sparse shale 3D texture.

In the following we discuss the transition arrow from 2D to 3D textures. The lack of spatial and frame-to-frame coherency and true 3D transparency can be resolved by creating 3D versions of the 2D textures. However the 3D textures can look different from their 2D originals depending on slice planes and projection angles. This means that property A is not satisfied. On the other hand, spatial coherency B_1 and frame-to-frame coherency B_2 is gained.

For a dense and opaque 3D texture property, C is not fulfilled. For a sparse 3D texture property, C is gained. Recognizing the original 2D texture might now be even more difficult than for a dense texture. The texture values accumulate in the depth direction during projection and clutter may arise due to the transparency. An example of this is shown in the top layer of Fig. 11.

Concerning the extrapolation of 2D textures to 3D dense textures, one option is to follow design guidelines to optimize the 3D texture for axis-aligned planar cuts. When restricting renderings to these cuts, the textures will have high recognizability. In this case property A is given at the expense of restrictions on how the volume can be sliced through.

The properties of using dense or sparse textures on a 3D proxy are shown on the respective arrows leading to the 3D proxy. Property A is negatively affected from both the extrapolation to 3D and from the sparse 3D texture representation.

In Fig. 11 we give an example of how the 2D shale and limestone textures defined in Fig. 7 may look like when transformed to 3D sparse and 3D dense textures. 3D textures might be better suited for communicating knowledge than 2D textures. With the example in Fig. 11 we briefly investigate this hypothesis. The top and middle layers in Fig. 11 represent

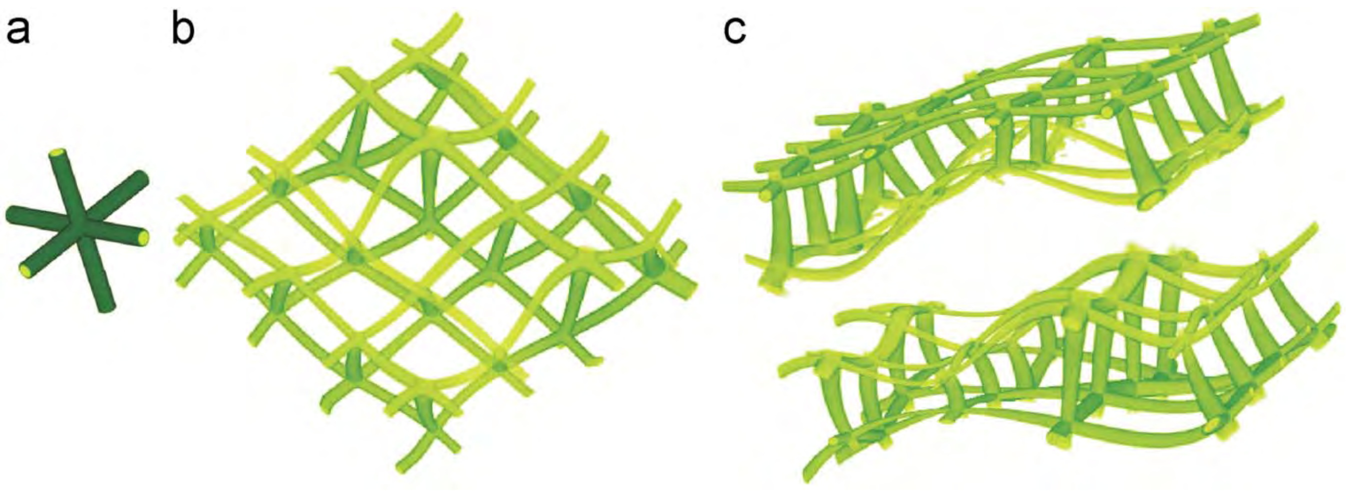


Fig. 12. (a) The texture primitive, (b) a deformed layer and (c) three deformed layers where the middle one is transparent.

limestone using a sparse and dense texture respectively. The dense limestone texture was designed with a blue core inside a black surrounding so that it will resemble the original 2D texture when sliced along axis aligned planes.

As an attempt at getting design guidelines for making the 3D textures we asked a geoscientific illustrator about the reasoning behind the design of the 2D seismic textures. The 2D textures stem from how the rock types look and break in nature and in borehole samples thus giving us concrete design guidelines for creating the 3D textures. Limestone will break in blocks due to its crystalline structure and shale is characterized by thin laminae. Therefore we extended the sparse limestone texture into blocks and the sparse shale texture was modeled with square sheets. We represented the green fill from the 2D shale texture as a highly transparent green haze and we made the square sheets black to preserve the colors from the 2D texture. With 3D textures the sparse top texture reveals the top surface of the middle layer. It allows insight into the data instead of only observing properties on the outer faces of the data as with 2D textures and 3D dense textures.

6.2. *Seismic knowledge in deformed textures*

Finally we discuss the effect of using deformed textures. Deformations carry knowledge as discussed in Section 5. If an oriented texture is used for a seismic layer, it communicates the orientation of the layer and the relative thickness of the layer through varying texture density. We have applied deformations on 3D sparse textures in Fig. 12. With a 3D sparse texture one can see the 3D deformation. Also the deformation throughout the whole layer is visible in one rendering. Obtaining this information with a 2D texture would require moving of a cut-plane through the volume. There are disadvantages of using 3D deformed textures as well. For example a sparse 3D texture does not show the deformation information with the same resolution as the 2D texture would. The problem of recognizability for deformed sparse textures is even more severe than for undeformed sparse textures.

7. Conclusions

We have presented the use of knowledge-assisted visualization through computer-assisted annotation for seismic interpretation. We have proposed to perform a top-down interpretation before the currently used bottom-up interpretation. This reduces the time for interpretation and for creating interactive communicative illustrations. Standardized textures used for annotating seismic data were presented. Their applicability in knowledge-assisted visualization was shown and the positive implications were discussed. Furthermore we discussed the advantages and disadvantages of extending 2D seismic textures to 3D. The work presented here is still an ongoing research. A larger project including funding has been initiated by an oil company for integrating these ideas into the daily workflow of oil and gas interpretation.

References

[1] Schlumberger information solutions (sis). Petrel, seismic interpretation software 〈http://www.slb.com/content/services/software/support/petrelcorner/〉; 2007.
[2] Bürger R, Hauser H. Visualization of multi-variate scientific data. In: EuroGraphics 2007 state of the art reports (STARs); 2007. p. 117–34.
[3] Castanie L, Levy B, Bosquet F. Volumeexplorer: roaming large volumes to couple visualization and data processing for oil and gas exploration. In: Proceedings of IEEE visualization '05; 2005. p. 247–54.
[4] Chen M, Ebert D, Hagen H, Laramee R, van Liere R, Ma K-L, et al. Data information and knowledge in visualization. In: IEEE visualization knowledge-assisted visualization workshop 2007.
[5] Committee FGD, editor. Federal geographic data committee, Digital cartographic standard for geological map symbolization. FGDC-STD-013-2006 〈www.fgdc.gov/standards/projects/FGDC-standards-projects/geo-symbol〉, 2006.
[6] Crawfis RA, Allison MJ. A scientific visualization synthesizer. In: VIS '91: proceedings of the 2nd conference on visualization '91, Los Alamitos, CA, USA: IEEE Computer Soc. Press; 1991. p. 262–7.
[7] Dong F, Clapworthy G. Volumetric texture synthesis for non-photorealistic volume rendering of medical data. The Visual Computer 2005;21(7):463–73.
[8] Emery D, Myers K, editors. Sequence stratigraphy. Oxford: Blackwell Scientific Publications; 2004.
[9] Faraklioti M, Petrou M. Horizon picking in 3D seismic data volumes. Machine Vision and Applications 2004;15(4):216–19.
[10] Grotzinger J, Jordan TH, Press F, Siever R. Understanding Earth. New York: Freeman; 1994.
[11] Iske A, Randen T, editors. Atlas of 3D seismic attributes, mathematics in industry, mathematical methods and modelling in hydrocarbon exploration and production. Berlin, Heidelberg: Springer; 2006.
[12] Kirby RM, Marmanis H, Laidlaw DH. Visualizing multivalued data from 2D incompressible flows using concepts from painting. In: Ebert D, Gross M, Hamann B, editors, IEEE visualization '99. San Francisco; 1999. p. 333–40.
[13] Kopf J, Fu C-W, Cohen-Or D, Deussen O, Lischinski D, Wong T-T. Solid texture synthesis from 2D exemplars. In: ACM transactions on graphics (Proceedings of SIGGRAPH 2007), 2007. p. 2:1–9.
[14] Lidal EM, Langeland T, Giertsen C, Grimsgaard J, Helland R. A decade of increased oil recovery in virtual reality. IEEE Computer Graphics and Applications 2007;27(6):94–7.
[15] Lu A, Ebert DS. Example-based volume illustrations. In: Proceedings of IEEE visualization 2005, 2005. p. 83–92.
[16] Lu A, Ebert DS, Qiao W, Kraus M, Mora B. Volume illustration using wang cubes. ACM Transactions on Graphics (TOG) 2007;26(2).

[17] Owada S, Nielsen F, Okabe M, Igarashi T. Volumetric illustration: designing 3D models with internal textures. In: Proceedings of the 2004 SIGGRAPH conference, 2004. p. 322–8.
[18] Patel D, Giertsen C, Thurmond J, Gjelberg J, Gröller ME. The seismic analyzer—interpreting and illustrating 2D seismic data. IEEE Transaction on Visualization and Computer Graphics 2008;14(6):1571–8.
[19] Patel D, Giertsen C, Thurmond J, Gröller ME. Illustrative rendering of seismic data. In: Lensch HPA, Rosenhahn B, editors. Proceeding of vision modeling and visualization 2007; November 2007. p. 13–22.
[20] Pepper R, Bejarano G. Advances in seismic fault interpretation automation. In: Search and Discovery article 40170, Poster presentation at AAPG annual convention, AA, 2005. p. 19–22.
[21] Plate J, Tirtasana M, Carmona R, Fröhlich B. Octreemizer: a hierarchical approach for interactive roaming through very large volumes. In: Proceedings of VISSYM '02; 2002. p. 53–64.
[22] Rautek P, Bruckner S, Gröller E. Illustrative visualization—new technology or useless tautology? Computer Graphics Quarterly, VisFiles 2008; 42(3).
[23] Ropinski T, Steinicke F, Hinrichs KH. Visual exploration of seismic volume datasets. In: Journal proceedings of WSCG '06, vol. 14, 2006. p. 73–80.
[24] Taylor RM. Visualizing multiple fields on the same surface. IEEE Computer Graphics and Applications 2002;22(3):6–10.
[25] Viola I, Sousa MC, Ebert D, Andrews B, Gooch B, Bruckner S, et al. Illustrative visualization for science and medicine full-day tutorial. In: IEEE visualization, 2006.
[26] Wang L, Mueller K. Generating subresolution detail in images and volumes using constrained texture synthesis. In: Proceedings of IEEE visualization 2004; 2004. p. 75–82.